BlueMod+S50/Central
AT Command Reference

80578ST10890A Rev. 2 – 2018-09-13

TELIT
TECHNICAL
DOCUMENTATION

SPECIFICATIONS ARE SUBJECT TO CHANGE WITHOUT NOTICE

NOTICE

While reasonable efforts have been made to assure the accuracy of this document, Telit assumes no liability resulting from any inaccuracies or omissions in this document, or from use of the information obtained herein. The information in this document has been carefully checked and is believed to be reliable. However, no responsibility is assumed for inaccuracies or omissions. Telit reserves the right to make changes to any products described herein and reserves the right to revise this document and to make changes from time to time in content hereof with no obligation to notify any person of revisions or changes. Telit does not assume any liability arising out of the application or use of any product, software, or circuit described herein; neither does it convey license under its patent rights or the rights of others.

It is possible that this publication may contain references to, or information about Telit products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that Telit intends to announce such Telit products, programming, or services in your country.

COPYRIGHTS

COMPUTER SOFTWARE COPYRIGHTS

USAGE AND DISCLOSURE RESTRICTIONS

I. License Agreements

The software described in this document is the property of Telit and its licensors. It is furnished by express license agreement only and may be used only in accordance with the terms of such an agreement.

II. Copyrighted Materials

Software and documentation are copyrighted materials. Making unauthorized copies is prohibited by law. No part of the software or documentation may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, without prior written permission of Telit.

III. High Risk Materials

Components, units, or third-party products used in the product described herein are NOT fault-tolerant and are NOT designed, manufactured, or intended for use as on-line control equipment in the following hazardous environments requiring fail-safe controls: the operation of Nuclear Facilities, Aircraft Navigation or Aircraft Communication Systems, Air Traffic Control, Life Support, or Weapons Systems (High Risk Activities"). Telit and its supplier(s) specifically disclaim any expressed or implied warranty of fitness for such High Risk Activities.

IV. Trademarks

TELIT and the Stylized T Logo are registered in Trademark Office. All other product or service names are the property of their respective owners.

V. Third Party Rights

The software may include Third Party Right software. In this case you agree to comply with all terms and conditions imposed on you in respect of such separate software. In addition to Third Party Terms, the disclaimer of warranty and limitation of liability provisions in this License shall apply to the Third Party Right software.

TELIT HEREBY DISCLAIMS ANY AND ALL WARRANTIES EXPRESS OR IMPLIED FROM ANY THIRD PARTIES REGARDING ANY SEPARATE FILES, ANY THIRD PARTY MATERIALS INCLUDED IN THE SOFTWARE, ANY THIRD PARTY MATERIALS FROM WHICH THE SOFTWARE IS DERIVED (COLLECTIVELY “OTHER CODE”), AND THE USE OF ANY OR ALL THE OTHER CODE IN CONNECTION WITH THE SOFTWARE, INCLUDING (WITHOUT LIMITATION) ANY WARRANTIES OF SATISFACTORY QUALITY OR FITNESS FOR A PARTICULAR PURPOSE.

NO THIRD PARTY LICENSORS OF OTHER CODE SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND WHETHER MADE UNDER CONTRACT, TORT OR OTHER LEGAL THEORY, ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE OTHER CODE OR THE EXERCISE OF ANY RIGHTS GRANTED UNDER EITHER OR BOTH THIS LICENSE AND THE LEGAL TERMS APPLICABLE TO ANY SEPARATE FILES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

APPLICABILITY TABLE

PRODUCTS

- BLUEMOD+S50/AI/CEN
- BLUEMOD+S50/AP/CEN
- BLUEEVA+S50/AI/CEN

CONTENTS

1. INTRODUCTION

1.1. Scope

This document specifies the command interface for the BlueMod+S50/Central firmware.

1.2. Audience

This document is intended for Telit customers, especially system integrators, about to implement Bluetooth modules in their application.

1.3. Contact and Support Information

For general contact, technical support services, technical questions and report documentation errors contact Telit Technical Support at:

- TS-SRD@telit.com

Alternatively, use:

https://www.telit.com/contact-us/

For detailed information about where you can buy the Telit modules or for recommendations on accessories and components visit:

https://www.telit.com

Our aim is to make this guide as helpful as possible. Keep us informed of your comments and suggestions for improvements.

Telit appreciates feedback from the users of our information.

1.4. Text Conventions

STOP Danger – This information MUST be followed or catastrophic equipment failure or bodily injury may occur.

⚠ Caution or Warning – Alerts the user to important points about integrating the module, if these points are not followed, the module and end user equipment may fail or malfunction.

ℹ Tip or Information – Provides advice and suggestions that may be useful when integrating the module.

All dates are in ISO 8601 format, i.e. YYYY-MM-DD.

1.5. Related Documents

- [1] BlueMod+S50 Hardware User Guide, 1VV0301505
- [2] BlueMod+S50 Software User Guide, 1VV0301506
- [3] Bluetooth 5.0 Core Specification
- [4] UICP+ UART Interface Control Protocol, 30507ST10756A

2. FEATURES

The BlueMod+S50/Central supports AT command mode and multiplexing mode via UART.

In the factory-default configuration the BlueMod+S50/Central is set to AT command mode using a UART baud rate of 115,200 bps, 8 data bits, no parity, 1 stop bit (8N1) and hardware flow control.

2.1. AT Command Mode

Each command line consists of a prefix, a body and a terminator.

All command lines begin with the prefix AT (ASCII 065, 084) or at (ASCII 097, 116).

The body is a string of characters in the ASCII range 032-255. Control characters other than <CR> (carriage return; ASCII 013) and <BS> (back space; ASCII 008) in a command line are ignored.

Note: The control characters are configurable via S registers.

• *Carriage return character (CR)*	*S3 register*
• *Line feed character (LF)*	*S4 register*
• *Back space character (BS)*	*S5 register*

The terminator is <CR>.

There is no distinction between upper-case and lower-case characters. A command line can have a maximum length of 80 characters. It is automatically discarded if the input is longer. Corrections are made using <BS>. Multiple commands on the same command line are not allowed.

Commands have the following syntax:

Syntax	Description
AT<command>=<value><CR>	Write the value of the command
AT<command>?<CR>	Read the value of the command

Exceptions of this syntax are marked separately.

Responses are sent back to the host and can be any of the following:

Responses	Description
<CR><LF>value<CR><LF>	Current value
<CR><LF> list entry 1<CR><LF> list entry 2<CR><LF> … list entry n<CR><LF> <CR><LF>	List value (e.g. AT+BNDLIST)
<CR><LF>OK<CR><LF>	Successful final message
<CR><LF>ERROR<CR><LF>	Error message, command not supported

2.2. Escape Sequence

To enter the command mode during an active data connection the following sequence (escape sequence) can be used:

<delay time [1] ><+><+><+><delay time>

The time interval between each of the three plus signs must not exceed 1 second. The escape sequence remains transparent to the remote device.

The escape sequence character is configurable via S2 register.

2.3. Multiplexing Mode (MUX)

*) *Supported since firmware version 5.1*

The multiplexing mode is used to handle incoming and outgoing data of different remote endpoints and command data.

Data has to be sent and are received in the following framing (all values in hexadecimal format):

Name	Description	Length	Value
Start	Start of frame	8 bit	CC
Channel ID	Channel identifier	8 bit	00 – FF
Length	Length of data	8 bit	-
Data	Max. 255 bytes data	Min. 0 byte Max. 255 bytes	-

The start byte is used to detect the start of a frame.

[1] Delay time defined in the S12 register (see page 27)

The channel ID determines the channel to send data to. This can be the data channel of a BLE GATT characteristic or the TIO data channel or the AT command interface (value FF).

The length field sets the length of the payload to send or received in bytes.

The data field consists of the payload data to send or receive.

Start of frame, channel ID, length and data are always transmitted in direct, binary form. AT commands have to be sent to the channel ID FF, simply prefixed with start of frame, FF, and length byte. Data received from the AT command interface are marked by channel ID FF. Line editing using backspace is not available in multiplexing mode.

Examples:

CC 01 0B 31 32 33 34 35 36 37 38 39 30 0D	Send data “1234567890<CR>” to channel 1 via MUX protocol
CC FF 06 41 54 49 39 39 0D	Send AT command “ATI99<CR>” via MUX protocol
CC FF 06 0D 0A 4F 4B 0D 0A	Receive response “<CR><LF>OK<CR><LF>” from AT command interface via MUX protocol

Result messages like RING, CONNECT and NO CARRIER may be sent in multiple frames by the BlueMod+S50/Central module. The host controller application needs to collect the data until the closing “<CR><LF>” is received.

2.4. Connection Establishment Procedure

The parameters AT+LECONINTMAX and AT+LESLAVELAT are used for central connection establishment. The connection supervision timeout used for connection establishment is calculated due to conform to the Bluetooth core spec..

On a peripheral, 200 ms after an incoming connection establishment the peripheral will check if the used connection parameters are compatible with the parameters AT+LECONINTMIN, AT+LECONINTMAX and AT+LESLAVELAT. If the parameters are not compatible, the peripheral will update the connection parameters automatically to the configured parameters. The connection supervision timeout used by automatic parameter update is calculated due to conform to the Bluetooth core spec.. If connection parameter update fails, the peripheral retries the connection parameter update. The maximum number of attempts is 3 and the time between the retries is 5 seconds.

2.5. Hangup

All active data connections will be disconnected at once by setting the HANGUP pin (GPIO4) to high level.

3. COMMAND SUMMARY

The description of the commands is structured into the following parts:

- General commands
- Bluetooth Low Energy (BLE) specific commands
- Generic Attribute Profile (GATT) specific commands

The factory-default values of the commands are marked using the bold letter format.

3.1. General

%B Baud Rate

AT syntax: **AT%B<value>**

This command determines the baud rate for the UART.

The following standard baud rates are supported:

Value	Description
4	9,600 bps
5	19,200 bps
6	38,400 bps
7	57,600 bps
8	115,200 bps **(default)**
9	230,400 bps
22	460,800 bps
23	921,600 bps

Additionally, to the standard baud rates described above, it is possible to set the following baud rates.

Value	Description
1200	1,200 bps
2400	2,400 bps
4800	4,800 bps
9600	9,600 bps
14400	14,400 bps
19200	19,200 bps
28800	28,800 bps
38400	38,400 bps
57600	57,600 bps
76800	76,800 bps
115200	115,200 bps
230400	230,400 bps
250000	250,000 bps
460800	460,800 bps
921600	921,600 bps
1000000	1,000,000 bps

To set a baud rate write the desired value in the command.

Examples:

AT%B9	Set baud rate to 230,400 bps
AT%B250000	Set baud rate to 250,000 bps

Information regarding the deviation of the real baud rate to the value set can be found in the *BlueMod+S50 Hardware User Guide [1]*.

&F Load Factory Defaults

AT syntax: **AT&F<value>**

The factory-default values will be loaded. For storing values in non-volatile memory, use the **AT&W** command.

Value	Description
0	Set all parameters except **bndlist** to factory defaults
1	Set all parameters to factory defaults

When **AT+LEPRIVACY=2** is active (Link Layer Privacy enabled) the **AT&F1** command generates a new Identity Resolving Key (IRK). This key is used to calculate the local random resolvable address and exchanged during bonding. By changing the IRK all previously bonded peer devices are no longer able to resolve the local random resolvable address.

Some restored values require an additional reset to get active (e.g. **AT+LETIO**, **AT+UICP**).

&W Store Active Configuration

AT syntax: **AT&W**

The active configuration is stored in non-volatile memory.

A Accept Incoming Call

AT syntax (normal mode): **ATA**
AT syntax (MUX mode): **ATA <channel ID>**

This command is supported for compatibility reasons only. A Bluetooth low energy device has to answer every connection automatically. See also register S0 description.

Examples:

ATA	Accepts connection in normal mode
ATA 0x01	Accepts connection in MUX mode at channel ID 0x01

+BIOCAP SSP I/O Capabilities

AT syntax: **AT+BIOCAP=<value>**

This command sets the input and output capabilities of the device used for SSP.

Value	Description	Related commands	Related events
0	Display only	n.a.	SSPPIN
1	Display Yes/No	AT+BSSPCONF (LE secure connection)	SSPPIN (LE legacy pairing) SSPCONF (LE secure connection)
2	Keyboard only	AT+BSSPPIN	SSPPIN
3	No input no output **(default)**	n.a.	n.a.
4	Display and keyboard	AT+BSSPPIN (LE legacy pairing) AT+BSSPCONF (LE secure connection)	SSPPIN (LE legacy pairing) SSPCONF (LE secure connection)

+BMITM SSP Man in the Middle Protection

AT syntax: **AT+BMITM=<value>**

This command controls the man in the middle (MITM) protection of the device during SSP. It has to be set in context with **AT+BIOCAP** command. In SSP there are scenarios where MITM protection is not possible.

Value	Description
0	Parameter disabled, connection and service based configuration applies (see **ATD** command and **AT+LETIO** parameter) **(default)**
1	Man in the middle protection enabled (connection and service based configuration is ignored)

In case the user chooses a scenario where MITM protection is not possible but one of the communication devices is configured to **AT+BMITM=1** (MITM protection enabled), the pairing is refused.

For possible combinations of I/O capabilities and the possibility of MITM protection/authentication level see command **AT+BIOCAP** and the *BlueMod+S50 Software User Guide [2]*.

+BMUX Activate Multiplexing Mode

AT syntax: **AT+BMUX=<value>**

*) *Supported since firmware version 5.1*

This command is used to activate the multiplexing mode protocol.

Value	Description
0	Normal AT mode **(default)**
1	Non persistent multiplexing mode
2	Persistent multiplexing mode

In the default configuration the device is working in normal AT mode. Setting AT+BMUX=1 enables the multiplexing mode.

ℹ After receiving “OK” in the response of the AT+BMUX=1 command all subsequent commands have to be entered in multiplexing frame format.

The multiplexing mode 1 is not stored persistent. To disable the multiplexing mode the device must be reset.

The multiplexing mode 2 is stored persistent. To disable the multiplexing mode the command AT+BMUX=0 followed by AT&W must be entered. This mode is intended to be used with the AT+SYSTEMOFF command.

In multiplexing mode extended result codes are always active (see chapter 4.3).

+BNAME Local Device Name

AT syntax: **AT+BNAME=<name>**

This command allows the modification of the local device name. The device name is shown on a remote Bluetooth device during device/service discovery. It is limited to 19 characters.

The device name can contain a format element to include the device’s own address or parts of it in the name.

In BLE advertising the name is truncated to the first 10 characters.

Format: “%[<s>][<d>]a”

“%”	Identifier start format element
<s>	Character separator on byte order (optional)
<d>	Number (1-12) of digits included in device name (optional, default is 4)
“a”	Identifier end format element

Examples: Device address = “0123456789AB”

AT+BNAME=BM+S50 %3a	Display on remote end: BM+S50 9AB **(default)**
AT+BNAME=BM+S50 %4a	Display on remote end: BM+S50 89AB
AT+BNAME=BM+S50 %:3a	Display on remote end: BM+S50 9:AB
AT+BNAME=BM+S50 %3a	Display on remote end: BM+S50 9AB
AT+BNAME=BM+S50 %:12a	Display on remote end: BM+S50 01:23:45:67:89:AB

+BNDDEL Delete Bonding Information

AT syntax: **AT+BNDDEL=<value>**

This command deletes the bonding information stored by the BlueMod+S50/Central.

Value	Description
Bluetooth address	Delete the bond of the device with specified address from the bonded devices list
*	Delete all bonded devices from the bonded devices list

This command can only be executed while not connected.

+BNDLIST Show Bonded Device List

AT syntax: **AT+BNDLIST**

This command shows information about the devices bonded with the BlueMod+S50/Central.

Each entry in the bonded devices list contains the Bluetooth address and the linktype (see chapter 4.2) and the role of the remote device (“C” for client or “P” for peripheral).

There may be exist two entries for one device if it supports client and peripheral role both.

Example:

AT+BNDLIST	0080254800DA 0x02 C 0080254800DA 0x02 P 9C04EB06ACA2 0x03 P OK

+BNDS Storage Mode for Bonds

AT syntax: **AT+BNDS=<mode>**

This command controls the storage mode for bonding information (link keys).

Mode	Description
0	Bonds persists for the duration of the authenticated connection
1	Bonds are permanently stored in the NVRAM of the BlueMod+S50/Central **(default)**

ℹ Modification of this parameter will delete all entries in the bonded devices list.
The bonding information is stored in the module flash. If your application does not need to store bonds switch this parameter to 0, to protect the module flash from unnecessary clear and write operations. Every flash has a limited number of clear cycles.

+BNDSIZE Bonded Devices List Size

AT syntax: **AT+BNDSIZE=<value>**

This command reduces the number of entries (1…4) the bonded devices list can hold.
The BlueMod+S50/Central can store up to 4 devices. The default size is **4**.

ℹ Reducing the parameter below the number of currently bonded devices will delete all entries in the bonded devices list.
Two entries of the same device are counted as one device.

Example:

AT+BNDSIZE=1	Limit the number of bonded devices to 1

+BOAD Bluetooth Own Device Address

AT syntax: **AT+BOAD**

This command reads the Bluetooth devices' own device address.

This command is read only.

+BPAIRMODE Configure Pairable Mode

AT syntax: **AT+BPAIRMODE=<mode>**

This command controls the pairable mode of the BlueMod+S50/Central peripheral part.

When set to "0" the module is only connectable for clients stored in the local bonded devices list. New pairing requests will be rejected.

Mode	Description
0	No pairing allowed, BlueMod+S50/Central advertises TIO as "functional"
1	Pairing allowed, BlueMod+S50/Central advertises TIO as "bondable and functional" **(default)**

This command restricts the access only to security enabled characteristics.

For Terminal I/O this means **AT+LETIO** has to be set to a value different than 0 or 2.

+BSSPCONF Security Pairing Confirmation

AT syntax: **AT+BSSPCONF <Bluetooth address>,<address type>,<value>**

If a pairing is initiated and LE secure connection is supported, depending on the security settings AT interface generates an event SSPCONF and asks the user for confirmation.

Event: SSPCONF <Bluetooth address>,<address type> <passkey> ?

The user has to confirm the passkey with the above command. If no confirmation is sent by the user within the bonding timeout or in case of active reject, the pairing is rejected with NO CARRIER message.

Value	Description
0	Reject passkey confirmation request
1	Accept passkey confirmation request

Example:

SSPCONF 00802507C08D,t2 794851 ? AT+BSSPCONF 00802507C08D,t2,1 OK RING CONNECT	Receive SSP pairing request Send SSP pairing confirmation

+BSSPPIN SSP Passkey Response

AT syntax: **AT+BSSPPIN <Bluetooth address>,<address type>, <SSP passkey>**

If an authentication is initiated, depending on the I/O capabilities (**AT+BIOCAP**) the AT interface generates an event SSPPIN and asks the user for the SSP passkey.

Event: SSPPIN <Bluetooth address>,<address type> ?

The user has to answer this request with the SSP passkey displayed on the remote device.

Example:

SSPPIN 00802507C08D,t2 ? AT+BSSPPIN 00802507C08D,t2,314546 OK RING CONNECT	Receive SSP passkey request Send SSP passkey response

+BSSPDBG SSP Debug Mode

AT syntax: **AT+BSSPDBG=<value>**

This command allows to enable the SSP debug mode of the device. This mode is required to trace the SSP Bluetooth connection by using a sniffer.

Value	Description
0	SSP Debug mode off **(default)**
1	SSP Debug mode on (use LESC debug key pair)
2	SSP Debug mode on (disable LESC)

⚠ SSP Debug mode is for tracing purposes only and shall be deactivated for normal operation. Otherwise the connection may be unsecure.

D Initiate Bluetooth Low Energy Link

AT syntax: **ATD<brad>[,tx],GATT[,reusebond]**
ATD<brad>[,tx],TIO[,reusebond]

This command addresses a Bluetooth device directly via its address.

Parameter	Description
brad	Called Bluetooth remote device address (12 hex digits)
tx	x is the remote Bluetooth address type (see chapter Bluetooth Address). If not specified a public address is assumed.
reusebond	Optional parameter that automatically starts the encryption immediately after the GATT connection is established. This is possible only if bonding information is available for the remote device. If no such bonding information is available this parameter has no effect. Note: It is not possible to read out the information of an encrypted connection.

If the device is configured to “Central” role and initiates a connection to a peripheral device, it shall use the identifier GATT or TIO.

A GATT connection allows to use the GATT Client specific command (see chapter 3.3).

A TIO connection allows to transfer data transparently for the UART to the peer device.

If no identifier is given in the dial string, no connection attempt will be initiated. The command reports ERROR.

Any character input while the BlueMod+S50/Central is dialing will cancel the dialing procedure.

Dialing procedure ends after a timeout specified by S register S7.

Dialing procedures which cause a security procedure have additional timeouts depending on the requested security procedure (see parameter **AT+BIOCAP**). The connection timeout specified in S7 is not valid after a security procedure is started.

Except for the <reusebond> parameter functionality described above all security procedures are expected to be requested by the remote device while connection setup or an active connection. In case the remote devices request a security procedure the BlueMod+S50/Central automatically performs all necessary procedures to satisfy the security needs of the remote device only limited by the remote and local I/O capabilities (**AT+BIOCAP**).

+DFUMODE Device Firmware Update Mode

AT syntax: **AT+DFUMODE=<value>**

This command sets the device firmware update mode.

To activate the mode it is necessary to store the settings and perform a reset or use the command **AT+DFUSTART**.

Value	Description
1	Device firmware update over serial interface **(default)**
2	Device firmware update over the air (OTA)

+DFUNAME Over The Air Update Name

AT syntax: **AT+DFUNAME=<name>**

This command sets the device name for the over the air firmware update mode.

The name is limited to 8 characters.

To activate the device name it is necessary to store the settings and perform a reset or use the command **AT+DFUSTART**.

Name	Description
BM+S_DFU	Device firmware update name is “BM+S_DFU” **(default)**

+DFUSTART Start Bootloader

AT syntax: **AT+DFUSTART**

This command sets the device into the configured firmware update mode. The command times out after 2 minutes.

E Local Echo

AT syntax: **ATE<value>**

This command selects the local echo in command mode.

Value	Description
0 or empty	No local echo
1	Local echo on in command phase **(default)**

H Disconnect

AT syntax: **ATH <connHnd>**

This command disconnects the existing Bluetooth connection addressed by connHnd from the corresponding CONNECT event.

Examples:

ATH 0x10	Disconnects connection with connHnd 0x10
ATH 0x01	Disconnects connection with connHnd 0x01

I Display Version Information

AT syntax: **ATI<value>**

Displays different information about version number and settings.

Value	Description
0 or empty	Returns the device name (e.g. “BM+S50 %3a“)
1	Returns “0”
2	Returns “OK”
3	Returns the version string: “V5.x.yyyy”
4	Returns the manufacturers name: “Telit”
5	Returns “ERROR”
6	Returns the copyright string: “(c) Copyright Telit”
7	Returns “OK”
8	Returns “ERROR”
9	Returns “OK”
77	Returns bootloader version
99	Returns the firmware creation date

+IOACFG Config of Pin IOA

AT syntax: **AT+IOACFG=<value>**

This command configures the function of the IOA pin (GPIO8). GPIO8 can be configured as output signal “Device Ready". The signal is low active, a low level shows the device ready to process commands and establish Bluetooth connections after startup.

Value	Description
0	Disconnected (no function, lowest power consumption)
1	Output: Signal “Device Ready” **(default)**

+IOBCFG Config of Pin IOB

AT syntax: **AT+IOBCFG=<value>**

This command configures the function of the IOB pin (GPIO3). GPIO3 can be configured as output signal "Data Carrier Detect" (DCD). The signal is low active. A low level signals that the device is connected on Terminal I/O level to a remote device.

Value	Description
0	Disconnected (no function, lowest power consumption) **(default)**
1	Output: Signal “DCD”
2	Output: Low level if a lower layer connection is active Output: High level if no lower layer connection is active

+LOAD Load Stored Parameter Setting

AT syntax: **AT+LOAD**

This command loads all parameters stored in non-volatile RAM.

+NFCMODE Set NFC Mode

AT syntax: **AT+NFCMODE=<value>**

This command sets the operation mode of the NFC interface.

The BlueMod+S50/Central provides the possibility to connect an NFC antenna directly to the module (refer to the *BlueMod+S50 Hardware User Guide [1]*).

Value	Description
0	NFC interface off **(default)**
1	Automatic mode

O Return to Online State

AT syntax: **ATO**

If the BlueMod+S50/Central is in command mode after issuing an escape sequence while a connection is active, **ATO** returns the BlueMod+S50/Central to data mode.

+PNPPID PnP Product ID

AT syntax: **AT+PNPPID=<value>**

This command sets the product ID provided in the device information service (DIS). The format is a 16 bit hex value. The default is 0xB01B (Telit product ID for BlueMod+S50/Central firmware).

+PNPPVER PnP Product Version

AT syntax: **AT+PNPPVER=<value>**

This command sets the product version provided in the device information service (DIS). The format is a 16 bit hex value. The default value is the version number of the particular Telit BlueMod+S50/Central firmware, e.g. 0x0500 for firmware version 5.0.

After setting the vendor ID (**AT+PNPVID**) to a different value than the default 0x008F the user has to set his own product version (otherwise the value 0x0500 will be used).

+PNPVID PnP Vendor ID

AT syntax: **AT+PNPVID=<value>**

This command sets the vendor ID provided in the device information service (DIS). The format is a 16 bit hex value. The default value is 0x008F (Telit vendor ID).

+PNPVSRC PnP Vendor ID Source

AT syntax: **AT+PNPVSRC=<value>**

This command sets the vendor ID source provided in the device information service (DIS).

Value	Description
1	Bluetooth SIG assigned company ID **(default)**
2	USB assigned company ID

Q Suppress Results

AT syntax: **ATQ<value>**

This command allows/suppresses result codes and messages.

Value	Description
0 or empty	Enable result messages after command input **(default)**
1	Suppress result messages after command input

+RESET Reset Device

AT syntax: **AT+RESET**

This command resets the whole functionality of the BlueMod+S50/Central by a forced hardware reset (like power off/on).

+RFMAXTXPWR Maximum Output Power

AT syntax: **AT+RFMAXTXPWR=<value>**

This command sets the maximum output power of the Bluetooth radio of the device. A changed value becomes active immediately.

Value	Description
-128	Use factory default maximum output power of +4 dBm **(default)**
+4	+4 dBm
+3	+3 dBm
0	0 dBm
-4	-4 dBm
-8	-8 dBm
-12	-12 dBm
-16	-16 dBm
-20	-20 dBm
-40	-40 dBm

All other values in the range of -127 to 127 could be set with this command as well, but the equal or next lower value from the power table will be set internally. Furthermore, the value will be set to a value amongst minimum and maximum output power value of the device.

Example:

AT+RFMAXTXPWR=0	The maximum output power will be set to 0 dBm

S AT S Register

AT syntax: **ATSx=<value>**

This command configures the S register settings.

Mode	Description
Sx?	Show the current setting of register Sx
Sx=1	Set register Sx to 1

AT command S register set (all values in decimal format):

Register	Value	Description
S0	**1**	Immediate call acceptance *Note: Setting of S0 only allow value 1 for BLE. In case of BLE connections always one RING is signalized and automatic call acceptance is set.*
S2	**43**	Character for escape sequence
S3	**13**	Carriage-return character
S4	**10**	Line-feed character

Register	Value	Description
S5	**08**	Backspace character
S7	**30**	Wait time for connection (in s). 0 means no timeout
S12	**100**	Delay time by using the escape sequence in 10 ms increments

+SYSTEMOFF Enter System Off Mode

AT syntax: **AT+SYSTEMOFF[=<value>]**

This command sets the module into low power mode during the time the module is not used.

When setting **AT+SYSTEMOFF** (or **AT+SYSTEMOFF=1**) the module will wake-up on GPIO activity. To achieve the lowest power consumption set **AT+SYSTEMOFF=2**. In this case the module will wake-up by RESET signal.

The host can use the IOA pin to monitor the system status.

Value	Description
1	Wake up by GPIO
2	Wake up by RESET signal

Possible use cases and an usage example are described in the *BlueMod+S50 Software User Guide [2]*.

+UICP Set UART Interface Control Protocol

AT syntax: **AT+UICP=<mode>**

This command sets the mode of the UART Interface Control Protocol (UICP, see *[4]*).

To activate UICP, it is necessary to store the settings and perform a reset.

Mode	Description
0	UICP off **(default)**
1	UICP on

The valid sequence to e.g. activate UICP is:

AT+UICP=1

AT&W

AT+RESET

V Result Message Format

AT syntax: **ATV<value>**

This command determines the format of the result messages.

Value	Description
0 or empty	Result message is presented numerically (followed by <CR>)
1	Result message is presented as text **(default)**

W Extended Result Codes

AT syntax: **ATW<value>**

This command enables/disables extended result codes.

Value	Description
0 or empty	Result message is presented without extended result codes **(default)**
1	Result message is presented with extended result codes (include error causes)

3.2. Bluetooth Low Energy

+LEFIXPIN Fix PIN for Pairing Procedure

AT syntax: **AT+LEFIXPIN=fixpin**

This command specifies a 6 digit fixpin, to be used for the security procedure. If this value has a length of 0 (no digit specified in command) "**AT+LEFIXPIN=**" a randomly generated PIN is used. The default is a fixpin with length 0. To use this feature see also commands **AT+LETIO**, **AT+BIOCAP**. The fixpin is only used when the local input/output capabilities are set to display (**AT+BIOCAP=0)**.

For further information see *BlueMod+S50 Software User Guide [2]*.

Parameter	Description
fixpin	6 digits pin value e.g. "000000" (digit 0..9 only) (default is 0 length pin "")

+LEADINTMAX Maximum Advertising Interval

AT syntax: **AT+LEADINTMAX=<value>**

This command configures the maximum advertising interval (in milliseconds) for a Bluetooth Low Energy Peripheral.

Value	Description
n=20…10240	Use maximum advertising interval of n ms
1280	Use maximum advertising interval of 1280 ms **(default)**

+LEADINTMIN Minimum Advertising Interval

AT syntax: **AT+LEADINTMIN=<value>**

This command is not used in the BlueMod+S50/Central. It is just provided for compatibility reasons. The used advertising interval is set by **AT+LEADINTMAX** parameter.

+LEATTMTUMAX Maximum ATT MTU

AT syntax: **AT+LEATTMTUMAX=<value>**

This command configures the maximum ATT MTU used on all Bluetooth Low Energy links.

Value	Description
0	Sets maximum ATT MTU to the implementation defined maximum **(default)**
23..158	Sets maximum ATT MTU to the given value

+LECHMAP Channel Map Handling

AT syntax: **AT+LECHMAP=<value>**

AT+LECHMAP?connHnd

The command **AT+LECHMAP** used with "=" sets the Channel Map to be used for all Central connections.

The command **AT+LECHMAP** used with "?" shows the active Channel Map used by the connection defined by connHnd.

Value	Description
0000000002 … 1FFFFFFFFF	The useable/used data channels of a connection displayed as a 40 bit hex string. The 3 MSBs must be zero, at least 2 bits must be set.
1FFFFFFFFF	Use all 37 data channels **(default)**

+LECONINTMAX Maximum Connection Interval

AT syntax: **AT+LECONINTMAX=<value>**

This command configures the maximum connection interval for a Bluetooth Low Energy connection. The unit is in 1.25 milliseconds timeslots.

Value	Description
n=6…3200	Use maximum connection interval of n * 1.25 ms
32	Use maximum connection interval of 40 ms **(default)**

Make sure that the value of AT+LECONINTMAX is higher or equal the value of AT+LECONINTMIN.

+LECONINTMIN Minimum Connection Interval

AT syntax: **AT+LECONINTMIN=<value>**

This command configures the minimum connection interval for a Bluetooth Low Energy connection. The unit is in 1.25 milliseconds timeslots.

Value	Description
n=6…3200	Use minimum connection interval of n * 1.25 ms
16	Use minimum connection interval of 20 ms **(default)**

ℹ Make sure that the value of AT+LECONINTMAX is higher or equal the value of AT+LECONINTMIN.

+LECONPARAM Connection Parameter Handling

AT syntax:

AT+LECONPARAM=connHnd,[connIntMin],connIntMax,slaveLat[,connTimeout]
AT+LECONPARAM?connHnd

The command “AT+LECONPARAM” used with “=” requests new connection parameters to be used for the connection defined by connHnd.

The command “AT+LECONPARAM” used with “?” shows the active connection parameters used by the connection defined by connHnd. A LECONPARAM event is generated containing the active connection parameters.

Parameter	Description
connHnd	Connection handle from CONNECT event
connIntMin	6…3200 minimum connection interval in steps of 1.25 ms (mandatory for peripheral)
connIntMax	6…3200 maximum connection interval in steps of 1.25 ms
slaveLat	0…499 connection intervals
connTimeout	Optional connection supervision timeout in steps of 10 ms. Will be calculated internally if not specified. Has to be calculated according to Bluetooth core spec.

The connection establishment procedure is described in chapter 2.3.

In central role the optional parameter connIntMin is used for negotiation procedure with the peripheral. If the central does not support this feature, it will report an error. In peripheral role the parameter connIntMin is mandatory.

The new connection parameters are signaled by event LECONPARAM.

The result is OK or ERROR.

+LECPEVENT Enable LECONPARAM Event Signaling

AT syntax: **AT+LECPEVENT=<enable>**

This command enables/disables the automatic signaling of LECONPARAM events.

enable	Description
0	Disable automatic LECONPARAM event signaling **(default)**
1	Enable automatic LECONPARAM event signaling

The result is OK or ERROR.

+LEPHY PHY Selection Parameter

AT syntax: **AT+LEPHY=<value>**
AT+LEPHY?connHnd

The command **AT+LEPHY** used with "=" sets the preferred PHY for use during the PHY Update Procedure that is started after a new BLE connection is established.

The command **AT+LEPHY** used with "?" shows the active PHY used by the connection defined by connHnd.

Value	Description
0	No PHY Update Procedure is initiated by the local device. If the peer device initiates a PHY Update Procedure, the preferred PHY of the peer device is used. (Note: only symmetric use of "LE 1M" and "LE 2M" PHY are supported)
1	A PHY Update Procedure is started after a new connection is established. The preferred PHY is set to "LE 1M" for both directions.
2	A PHY Update Procedure is started after a new connection is established. The preferred PHY is set to "LE 2M" for both directions. **(default)**

+LEPRIVACY Enable Link Layer Privacy

AT syntax: **AT+LEPRIVACY=<value>**

This command controls the signaling of LERESOLVED events and the use of Link Layer Privacy for new BLE connections.

Value	Description
0	No LERESOLVED events are signaled during pairing. The local device uses its public address for all Bluetooth roles. **(default)**
1	When pairing with a peer device that is using a resolvable random address a LERESOLVED event is signaled. When scanning (**AT+LESCAN**) for a bonded peer device that is using a resolvable random address, the displayed address is the public address from the bond database. When connecting (**ATDxxx**) to a bonded peer device that is using a resolvable random address, the public address from the bond database can be used. The local device uses its public Bluetooth address for all Bluetooth roles.
2	Same as 1. The local device uses a random resolvable address for all Bluetooth roles. This address is changed every 15 minutes.

+LEROLE Bluetooth Low Energy Device Role

AT syntax: **AT+LEROLE=<value>**

This command configures the Bluetooth Low Energy role of the device.

Value	Description
0	Set device role to “Peripheral”
1	Set device role to “Central”
2	Set device role to combined “Peripheral” and “Central” **(default)**

When set to “Peripheral”, the device advertises and accepts incoming BLE connections. Searching for Bluetooth Low Energy devices with **AT+LESCAN** command is not possible.

When set to “Central”, the device is invisible and does not accept incoming BLE connections. The device can search for peripherals using **AT+LESCAN** command and initiate outgoing connections using the **ATD** command.

When set to the combined “Peripheral” and “Central” role (default) all above features are supported and have to be handled. If you only need one single role, please use parameter 0 or 1.

+LESLAVELAT Slave Latency

AT syntax: **AT+LESLAVELAT=<value>**

This command configures the slave latency (in connection intervals) for a Bluetooth Low Energy connection.

Value	Description
n=0…499	Use a slave latency of *n* connection intervals
0	Use no slave latency **(default)**

+LETIO Enable Terminal I/O Service

AT syntax: **AT+LETIO=<value>**

This command controls the Terminal I/O service. If set to 0 the Terminal I/O service is disabled. To activate the change, it is necessary to store the settings (**AT&W**) and perform a reset (**AT+RESET**).

Value	Description
0	Terminal I/O service disabled (no advertising, no characteristics)
1	Terminal I/O service enabled, security is required with encryption (no MITM)
2	Terminal I/O service enabled, no security (authentication or encryption) required **(default)**
3	Terminal I/O service enabled, authenticated pairing with encryption (MITM required)
4	Terminal I/O service enabled, authenticated LE secure connections pairing with encryption (MITM required, LE secure connections required)

The valid sequence to change the setting is:

AT+LETIO=1

AT&W

AT+RESET

+LEADDATA Setup Advertise Data for Customized Advertising

AT syntax: **AT+LEADDATA=<value$_1$> .. <value$_k$>**

This command is used to setup the advertise data for a customized advertising.

Value$_k$ represents an octet in hexadecimal format, k <= 31.

The coding of the data is according to the *Bluetooth 5.0 Core Specification / Vol. 3, Part C, Chapter 11 [3]*.

The “flags” data type must be present in the advertising data.

Example:

AT+LEADDATA=02010603020F18	Set flags (0x06) + UUID of battery service (0x180F)

+LEADE Enable Customized Advertising

AT syntax: **AT+LEADE=<value>**

This command controls the advertising behavior.

With **AT+LEADE=0** only the build in Terminal I/O service is advertised.

With **AT+LEADE=1** only the customized advertising value is advertised.

With **AT+LEADE=3** the module stops all advertising. With disabled advertising the client (e.g. iPhone) is not able to discover the device or to connect to the device. This should only be done when the service is not in use to save battery power.

With disabled internal TIO due to **AT+LETIO=0**, the values 0 and 3 show the same behavior. There will be no advertising and no connection.

Value	Description
0	Customized advertising disabled, internal TIO advertising enabled **(default)**
1	Customized advertising enabled, internal TIO advertising disabled
2	Reserved for future use
3	Advertising off, customized advertising disabled, internal TIO advertising disabled

+LEADPAR Setup Parameters for Customized Advertising

AT syntax: **AT+LEADPAR=par$_1$=<value$_1$>[, .. [,par$_n$=<value$_k$>]]**

This command is used to setup parameters for a customized advertising.

par$_n$	value$_k$
advtype Optional Coding: decimal.	Type of advertising: **0: undirected (default)** 2: scannable 3: non-connectable

This command is optional, if not submitted these defaults apply:

- advtype = 0 - advertising type “undirected”

The advertising interval may be set with the AT+LEADINTMAX command.

Example:

AT+LEADPAR=ADVTYPE=0	Set type of advertising “undirected”

+LESCDATA Setup Scan Response Data for Customized Advertising

AT syntax: **AT+LESCDATA=<value$_1$> .. <value$_k$>**

This command is used to setup the scan response data for a customized advertising.

Value$_k$ represents an octet in hexadecimal format, k <= 31.

The coding of the data is according to the *Bluetooth 5.0 Core Specification / Vol. 3, Part C, Chapter 11 [3]*.

The “flags” data type must be not present in the scan response data.

Example:

AT+LESCDATA=03020F18	Set UUID of battery service (0x180F)

3.3. GATT

3.3.1. GATT Client Specific AT Mode Commands

All commands described in this chapter can only be used in AT mode or in the AT channel in multiplexing mode (**AT+BMUX=1**).

+LEBUUIDSET Set 128 bit Base UUID

AT syntax: **AT+LEBUUIDSET=ux**

This command configures base UUIDs needed by the stack to identify 128 bit UUIDs correct.

128 bit UUIDs are module internally treated as 16 bit UUIDs with a defined base UUID. The Telit defined base UUID for Terminal IO V2 is 0000xxxx0000100080000008025000000 with xxxx as the variable 16 bit UUID part.

To set a base UUID the 16 bit UUID part could have any legal 16 bit value. It is internally ignored for this command. If the internal base UUID table is full the command reports ERROR, otherwise it reports OK.

The UUID list shall be saved permanent with **AT&W**.

Parameter	Description
ux	With x= 128 bit base UUID

+LEBUUIDDEL Delete Base UUID

AT syntax: **AT+LEBUUIDDEL=<value>**

This command deletes the base UUID information stored by the BlueMod+S50/Central in RAM.

To delete the UUID permanently from the flash, it is required to save the reduced list by **AT&W**, wait for OK and perform an **AT+RESET** command.

Value	Description
ux	With x= 128 bit base UUID
*	Delete all base UUIDs from the base UUID table

+LEBUUIDLIST Show Base UUID List

AT syntax: **AT+LEBUUIDLIST**

This command shows information about the configured 128 bit base UUIDs within the BlueMod+S50/Central.

The variable 16 bit UUID part is marked with the character ‘x’.

Example:

AT+LEBUUIDLIST	0000xxxx000010008000008025000000 0000xxxx111100002222008033330000 OK

+LEGATTEVENT Enable Additional GATT Events

AT syntax: **AT+LEGATTEVENT=<value>**

*) *Supported since firmware version 5.2*

With this command additional GATT events can be configured.

When enabled all incoming and outgoing connections are signaled with “CONNECT GATT” and “NO CARRIER” events. Also, a LEATTMTU event is signaled after MTU negotiation is completed.

Value	Description
0	Disable additional GATT events **(default)**
1	Signal all incoming and outgoing connections

+LESCAN Search Bluetooth Low Energy Devices

AT syntax: **AT+LESCAN**
AT+LESCAN=GATT[,<rssix>][,RAW]
AT+LESCAN=<ux>[,<rssix>][,RAW]
AT+LESCAN=<brad>[,<tx>][,<rssix>][,RAW]

With this command an automatic search for all discoverable Bluetooth Low Energy devices will be initiated. The discovery will last for a time defined by command **AT+LESCANDURATION**.

Parameter	Description
rssix	Filter for devices with RSSI value same or stronger x
brad	Filter for Bluetooth remote device address (12 hex digits)
tx	x is the remote Bluetooth address type see chapter LinkType. If not specified a public address is assumed
ux	With x=UUID of a service (4 or 32 hex digits)
GATT	Show all found devices
RAW	Hexdump of advertising and scan response data without duplicate filtering

Any character input while the BlueMod+S50/Central is searching will abort the search procedure.

The resulting list depends on the used command parameters.

As a result, a list will be output containing the Bluetooth addresses of the visible devices in range, the advertisement type, the RSSI, the Bluetooth friendly name, the TX level, manufacturer specific data and all UUIDs contained in the advertising and scan response data, if available, of the remote device. Please note that more AD types could be decoded in future releases.

Bluetooth address, RSSI and TYPE are always provided. All other values like Bluetooth friendly name, TX level, manufacturer specific data and UUID are optional and depends of the advertising data of the discovered device. UUIDs can be 16-bit or 128-bit values.

The output is filtered to show each unique advertising packet only once. If the internal filter table is full, all new advertising packets are shown unfiltered. The output in RAW mode is also unfiltered.

The optional parameter <rssi> (8-bit signed value) can be used to perform a search only for devices with a rssi value higher than the provided value. E.g. **AT+LESCAN=GATT,rssi-50** will show all devices with a rssi value same or higher than –50dBm. This means -45dBM devices are shown, -55dBm devices are filtered.

The optional parameter <brad> (12 hex digits) can be used to perform a search for a device with the specified Bluetooth address. The optional parameter <tx> specifies the type of Bluetooth low energy address.

The optional parameter <ux> (16-bit or 128-bit uuid value) can be used to perform a search for devices which advertises a specific service.

To show devices supporting Terminal I/O only, the UUID FEFB shall be used (**AT+LESCAN=uFEFB**).

If the “RAW” parameter is given, the output will not contain decoded AD type data. Instead it will contain the Bluetooth address of the visible device in range, the RSSI, the advertisement type and the complete advertise or scan response data from the remote device. The data is displayed as an ascii coded byte steam in hexadecimal values.

There will be no duplicate filtering for advertising packets. All received packets (advertise or scan response) will be printed as soon they are received. If UUID filtering is on (**AT+LESCAN=ux,RAW**), the output will be printed after receiving the complete advertising data. In active scan mode these are the advertising and scan response packets.

Examples:

AT+LESCAN	008025497826,t2 RSSI:-62 TYPE:CONN NAME:BM+SR 7 TX:4 MNF:8F0009B0011000 UUID:53544D544552494F5345525631303030 UUID:FEFB OK
AT+LESCAN=GATT,RAW	0080254800DD,t2 RSSI:-77 TYPE:CONN DATA:02010608FF8F0009B0011000 0080254800DD RSSI:-79 TYPE:SCANRSP DATA:110730303031565245534F495245544D54530302FBFE080 86A75657267656E OK

+LESCANDURATION Duration for +LESCAN

AT syntax: **AT+LESCANDURATION=<value>**

This command configures how long the BlueMod+S50/Central is searching for discoverable Bluetooth Low Energy devices when the command **AT+LESCAN** is used.

Value	Description
0	Sets duration time to infinite
1..300	Sets duration time between 1 seconds and 300 seconds (**default=10**)

+LESRVD Service Discovery

AT syntax: **AT+LESRVD=connHnd[,ux]**

With this command an automatic search for services on the given connection handle will be initiated. A connection is required before using this command.

Parameter	Description
connHnd	Connection handle from CONNECT event
ux	UUID of a service (4 or 32 hex digits)

The resulting list depends on the used command parameters.

AT+LESRVD=connHnd discovers all services.

Response: List of found services UUIDs.

AT+LESRVD=connHnd,ux discovers all characteristics for the given service UUID.

Response: Requested service UUID and a list of found characteristics with value handle, properties and UUIDs.

The first 4 bytes represent the hex coded **charHnd** value for this characteristic which has to be used with the other AT commands like **AT+LEREAD**.

The result parameter PROP describes the characteristic properties. They are coded as a hexadecimal bitmask as defined in *Bluetooth Core Spec 5.0, Vol. 3, Part G, Chapter 3.3.1.1 [3]*.

PROP	Properties
0x02	Read
0x04	Write without response
0x08	Write
0x10	Notify
0x20	Indicate

A characteristic with properties READ and WRITE reports “PROP:0x0A”.

The result parameter UUID shows the 16 bit or 128 bit hexadecimal UUID value of the found characteristic.

Example: Discover all services

AT+LESRVD=connHnd	UUID:1800 UUID:1801 UUID:180A UUID:FEFB OK

Example: Discover TIO service

AT+LESRVD=connHnd,uFEFB	UUID:FEFB 0x0015 PROP:0x04 UUID:00000001000010008000008025000000 0x0016 PROP:0x10 UUID:00000002000010008000008025000000 0x0018 PROP:0x08 UUID:00000003000010008000008025000000 0x0019 PROP:0x20 UUID:00000004000010008000008025000000 OK

+LEREAD Read Characteristic

AT syntax: **AT+LEREAD=connHnd,charHnd[,length]**

With this command a read access to the characteristic defined by connHnd and charHnd is initiated.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
length *)	Maximum data length to be read from Characteristic (optional)

*) *Parameter supported since firmware version 5.2*

The read data is displayed as ascii coded byte stream in hexadecimal values with a leading event identifier "LEREAD":

```
AT+LEREAD=connHnd,charHnd
LEREAD:connHnd,charHnd,<hexData>          e.g. LEREAD:0x10,0x0016,017AFF for three byte value
OK
```

For a longer characteristic value one or more "LEREADCONT" events are sent before a single "LEREAD" event:

```
AT+LEREAD=connHnd,charHnd
LEREADCONT:connHnd,charHnd,<hexData>          22 bytes of value
LEREADCONT:connHnd,charHnd,<hexData>          22 bytes of value
LEREAD:connHnd,charHnd,<hexData>              0 – 22 bytes of value
OK
```

+LEWRITE Write Characteristic

AT syntax: **AT+LEWRITE=connHnd,charHnd,<hexData>**

With this command a write with response access to the characteristic defined by connHnd and charHnd is initiated.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values, e.g. 017aFF for a three byte value. Valid value length: 0 – 20 bytes.

After receiving the response from the GATT server, depending on the result code the result is OK or ERROR. Also unknown values for connHnd and charHnd will lead to an ERROR result.

The host application has to provide the correct number of data bytes for the addressed characteristic. This length information could be found in the profile/service specification for the addressed service.

To initiate a write with response access with a larger value than 20 bytes the AT+LEWRITECONT command can be used.

+LEWRITECONT Write Characteristic Continue

AT syntax: **AT+LEWRITECONT=connHnd,charHnd,<hexData>**

*) *Supported since firmware version 5.2*

With this command a write with response access to the characteristic defined by connHnd and charHnd with a data length > 20 byte is prepared.

This command allows to fill the write buffer up to (ATT MTU – 3) bytes (see LEATTMTU).

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values, e.g. 017aFF for a three byte value. Valid value length: 20 bytes.

To write a large characteristic value the data is prepared with one or multiple AT+LEWRITECONT commands, while the actual write is triggered by a single AT+LEWRITE command:

AT+LEWRITECONT=<connHnd>,<charHnd>,<hexData>

OK

AT+LEWRITECONT=<connHnd>,<charHnd>,<hexData>

OK

AT+LEWRITE=<connHnd>,<charHnd>,<hexData>

OK

+LEWRITECMD Write without Response Characteristic

AT syntax: **AT+LEWRITECMD=connHnd,charHnd,<hexData>**

With this command a write without response (write command) access to the characteristic defined by connHnd and charHnd is initiated.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values, e.g. 017aFF for a three byte value. Valid value length: 0 – 20 bytes.

There is no result from the server side available. The result is OK if the write without response was sent to the server side. An ERROR is reported if sending was not possible. The result code does not contain information about the reception on the GATT server side.

To initiate a write without response access with a larger value than 20 bytes the AT+LEWRITECMDCONT command can be used.

+LEWRITECMDCONT Write without Response Characteristic Continue

AT syntax: **AT+LEWRITECMDCONT=connHnd,charHnd,<hexData>**

**) Supported since firmware version 5.2*

With this command a write without response (write command) access to the characteristic defined by connHnd and charHnd with a data length > 20 byte is prepared.

This command allows to fill the write buffer up to (ATT MTU – 3) bytes (see LEATTMTU).

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values. Valid value length: 20 bytes

To write a large characteristic value the data is prepared with one or multiple AT+LEWRITECMDCONT commands, while the actual write is triggered by a single AT+LEWRITECMD command:

```
AT+LEWRITECMDCONT=<connHnd>,<charHnd>,<hexData>

OK

AT+LEWRITECMDCONT=<connHnd>,<charHnd>,<hexData>

OK

AT+LEWRITECMD=<connHnd>,<charHnd>,<hexData>

OK
```

+LECCCD Enable/Disable Indications/Notifications

AT syntax: **AT+LECCCD=connHnd,charHnd,enable**

With this command the notifications and indications for the characteristic defined by connHnd and charHnd are enabled or disabled.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
enable	0: disable 1: enable notifications 2: enable indications

The result is OK or ERROR.

3.3.2. GATT Client Specific MUX Commands

All commands described in this chapter can only be used in multiplexing mode (**AT+BMUX=1**).

+LEADDCHAN Add MUX Channel to Connection

AT syntax: **AT+LEADDCHAN=connHnd,charHnd,writeType**

*) *Supported since firmware version 5.1*

This command adds a MUX channel for characteristic defined by charHnd to the connection defined by connHnd. All data transfer for this connection is now done using the MUX channel. A read is performed by issuing an AT+LEREADCHAN command on the AT channel. The data will be transferred using the corresponding MUX channel. All error conditions are signaled on the AT channel.

All data received within an indication or notification message is signaled on the corresponding MUX channel in a single MUX frame.

All data send to the MUX channel in a single MUX frame is send out as a write request. The type of request used by this channel is defined by the parameter “writeType”.

You can define only one channel per connection for a charHnd.

The commands AT+LEREAD, AT+LEWRITE and AT+LEWRITECMD respond ERROR when used with a charHnd with an active MUX channel.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from AT+LESRVD
writeType	“RSP”: Write with response “CMD”: Write without response

In case of successful MUX channel creation, a one byte hexadecimal coded MUX channel Id is signaled on a single line followed by the result code OK.

```
Format:     <CR><LF><channel><CR><LF>
            <CR><LF>OK<CR><LF>

E.g.        <CR><LF>0x17<CR><LF>
            <CR><LF>OK<CR><LF>
```

In case of an error condition the result code ERROR is signaled.

For more information on the MUX format see chapter 2.3.

+LEREADCHAN Read MUX Channel

AT syntax: **AT+LEREADCHAN=channel,[len]**

*) *Supported since firmware version 5.1*

With this command a read access to the characteristic addressed by a MUX channel is initiated.

Parameter	Description
channel	MUX channel Id from AT+LEADDCHAN
len	Length of characteristic data to be read

When using parameter len, the Host application has to provide the correct number of data bytes for the addressed characteristic. This length information could be found in the profile/service specification for the addressed service.

The read data is displayed in the corresponding MUX channel. The command responds OK or ERROR in the AT command channel.

For more information on the MUX format see chapter 2.3.

AT+LEREADCHAN=23

OK -> in AT channel

Data will be received in MUX channel 23.

3.3.3. GATT Client Data Handling on MUX Channel

MUX channels are created using the **AT+LEREADCHAN** command. MUX channels are valid during the connection.

All data for defined characteristics is sent in one MUX frame on the corresponding channel so a MUX frame has to have the defined data length for the characteristic addressed. If a characteristic has a length of 4, 4 bytes of data has to be sent. The only exception is a characteristic with a variable length.

3.3.3.1. Error Handling

In MUX mode all recognized errors are signaled on the AT command channel using the event LEERROR.

3.4. GATT

3.4.1. GATT Server Definition

+LEATTRIB	Define Attributes for a Service

AT syntax: **AT+LEATTRIB=<type>[,par$_1$=<value$_1$>[, .. ,par$_n$=<value$_n$>]..]]**

*) *Supported since firmware version 5.2*

This command is used to define attributes for one or more services in the GATT server. The maximum number of services and characteristics depends on the used features. Every combination results in a different number of possible service and characteristic combinations. The maximum number of possible characteristics is limited to 20. This restriction results from the limited size of the internal definition array. If the space is completely used the command “AT+LEATTRIB=charval,…” returns “ERROR”.

The GAP and GATT services that each GATT server must expose are built-in services and thus shall not be defined by the application!

The presence of parameters *par₁,..,parₙ* depends on the value of <*type*>:

type	par$_1$	par$_2$	par$_3$	par$_4$
pserv Mandatory	uuid=<*16/128bit UUID*> Mandatory Coding: hexadecimal.			
char Mandatory	prop=<*properties*> Mandatory Coding: hexadecimal. *properties* may have the bitmask values *Read*, *Write Without Response*, *Write*, *Notify* and *Indicate* set. Note: internally the controller generates an additional Client Characteristic Configuration Descriptor (CCCD) attribute with permissions „readable and writable without authentication or authorization“ if bits *Notify* or *Indicate* are set.			
charval Mandatory	uuid=<*16/128bit UUID*> Mandatory Coding: hexadecimal.	perm=<*permissions*> Mandatory Coding: hexadecimal. 16-bit value that decodes the access permissions and authentication requirements.	len=<*length*> OR vlen=<*length*> Mandatory Coding: decimal. Use “len” for fixed sized characteristic value or “vlen” for variable sized characteristic value. Maximum supported length value is 158 bytes.	val=<*value*> Optional Coding: hexadecimal. Pre-defined characteristic value that will be loaded when the service set is activated. Maximum supported length is 20 bytes.

type	**par$_1$**	**par$_2$**	**par$_3$**	**par$_4$**
chardcccd Optional	perm=<*permissions*> Mandatory Coding: hexadecimal. This command is needed only if a CCCD shall be generated with permissions other than „readable and writable without authentication or authorization“. See note in *type*=char description. CCCDs are required to be readable without authentication and required to be writable.			
chardusrd Optional	usrd=<*user description*> Mandatory Coding: UTF-8 string. User description string. Note: - Since *Extended Properties* (see *type*=char, parameter *properties* description) are not supported the remote peer may not write to this characteristic descriptor. - The number of characters is limited to 13. Internally the character sequence is terminated by a zero byte resulting in an ASCII-Z string.	perm=<*permissions*> Optional Coding: hexadecimal. 16-bit value that decodes the access permissions and authentication requirements. User Descriptions are required to be readable and required to be not writeable. If this parameter is omitted the default *readable without authentication or authorization* applies.		
complete Mandatory	No parameter. Used to signal that all attribute definitions have been sent to the controller.			

The characteristic properties are coded as a hexadecimal bitmask as defined in *Bluetooth Core Spec 5.0 Volume3 Part G Chapter 3.3.1.1 [3]*.

Value	Properties
02	Read
04	Write without response
08	Write
10	Notify
20	Indicate

The values can be combined, for example read & notify result in 12.

The attribute permissions (parameter perm=<*permissions*>) are bit coded in a 16 bit hexadecimal value.

Bit	Value	Function
0..3	0 1 2 3 4	Read not permitted Read permitted Read with authentication permitted Read with authentication and MITM protection permitted Read with authentication and LESC MITM protection permitted
4..7	0 1 2 3 4	Write not permitted Write permitted Write with authentication permitted Write with authentication and MITM protection permitted Write with authentication and LESC MITM protection permitted
8..15	-	Reserved

Example: Value for read only is 0001.

The complete service/s is/are defined through repeated submissions of the **AT+LEATTRIB** command (see example below).

The **AT+LEATTRIB** commands must be submitted in a specific order:

```
Definition of first service:
AT+LEATTRIB=pserv, …
    Definition of first characteristic of first service:
    AT+LEATTRIB=char, …
    … optional characteristic descriptors (AT+LEATTRIB=chardxxx) …
    AT+LEATTRIB=charval, …

    Definition of second characteristic of first service:
    AT+LEATTRIB=char, …
    … optional characteristic descriptors (AT+LEATTRIB=chardxxx) …
    AT+LEATTRIB=charval, …
    …
Definition of second service:
AT+LEATTRIB=pserv, …
    Definition of first characteristic of second service:
    AT+LEATTRIB=char, …
    … optional characteristic descriptors (AT+LEATTRIB=chardxxx) …
    AT+LEATTRIB=charval, …
    …
…
Completion of service and characteristics definition:
AT+LEATTRIB=complete
```

Upon successful execution of the **AT+LEATTRIB** command with *type=charval* the device returns the data channel ID which is associated to the characteristic value.

Format: <CR><LF>0x<*channel*><CR><LF>

If MUX mode is disabled (AT+BMUX=0) data belonging to the characteristic value is exchanged with the AT+LESRVDATA command / LESRVDATA event.

If MUX mode is enabled (AT+BMUX=1/2) data belonging to the characteristic value over multiplexer data channel ID *channel*. The channel value is coded in hexadecimal digits.

Example:

The example below shows the Battery Service. Battery Service is a simple service which exposes the battery charging level as single characteristic value.

Command	Response	Description
AT+LEATTRIB=pserv,uuid=180F AT+LEATTRIB=char,prop=12	OK OK	Declares the properties of the battery level value
AT+LEATTRIB=charval,uuid=2A19,perm=0001,len=1	0x20 OK	Declares the battery level value (one byte in the range 0,..,100). Battery level values are exchanged over channel 0x20
AT+LEATTRIB=complete	OK	Completes the service definition sequence

+LESRVSETOPEN Open a Service Set for Definition

AT syntax: **AT+LESRVSETOPEN=<value>**

*) *Supported since firmware version 5.2*

This command allows to define service sets. At the moment only 1 service set is supported.

The **AT+LESRVSETOPEN** command is only accepted when no service was previously activated with the **AT+LESRVSETACT** command or defined with **AT+LEATTRIB**.

The service set content is defined with the **AT+LEATTRIB** command, can be persistently stored with the **AT+LESRVSETSAVE** command.

The activation depends on the parameter **AT+LESRVBOOTMODE**. If **AT+LESRVBOOTMODE** is set to 0 the service set is activated with the **AT+LESRVSETACT** command.

If the **AT+LESRVSETOPEN** command is used to define a service set that exists already in flash memory the existing service is deleted from flash.

If the **AT+LESRVSETOPEN** command is submitted with parameter value = 0 a test mode is entered: services and characteristics can be defined without storing these in the flash memory. Each **AT+LEATTRIB** command is mapped to an operation on the Nordic SoftDevice API and the command parameters are immediately checked for integrity. The behavior is different for a non-zero parameter value: the parameters of the **AT+LEATTRIB** command are written to flash without preceding check for integrity. The check is made when the service set is activated with the **AT+LESRVSETACT** command. The reason is due to the fact the SoftDevice cannot remove registered services, defining more than one service set would require a target reset before a new service can be defined.

AT+LESRVSETOPEN with parameter value = 0 can be used to "develop" resp. "debug" a new service. Once it is found to be OK it can be then re-defined and stored with the validated command sequence for a non-zero service set.

Value	Description
n=0	Select dummy service set
n=1	Select service set n for new service definitions

+LESRVSETSAVE Save a Service Set Definition

AT syntax: **AT+LESRVSETSAVE**

*) *Supported since firmware version 5.2*

This command persistently stores the service set definition that was previously addressed with the **AT+LESRVSETOPEN** command and returns a 16 bit checksum for the activated set as human readable 0xXXXX string.

The checksum is calculated "live" at save time and can be used as data consistency indicator while later activation of a set since the checksum of the set activation is also calculated live while activation and shall result in the same value in case the set data is not corrupted due to FLASH or RAM failure.

+LESRVSETACT Activates a Service Set

AT syntax: **AT+LESRVSETACT=<value>**

*) *Supported since firmware version 5.2*

This command allows to activate a service set stored in the FLASH.

In case any service set is already activated, the command will respond with "ERROR".

In case the definitions of the activated services are detected as not consistent at GATT level, the command will respond with "ERROR".

In case the definitions of the activated services are successfully enabled, the command will return a 16 bit checksum for the activated set as human readable 0xXXXX string followed by "OK".

That checksum is calculated "live" at activation time and can be used as data consistency indicator while later activation of a set since the checksum of the set save operation was also calculated live while storage and shall result in the same value in case the set data is not corrupted due to FLASH or RAM failure.

Value	Description
n=1	Select service set n for activation. The BlueMod+S50 will make all service information that are defined for the activated service set available via BLE and create one channel ID for each characteristic value of the service set definition

+LESRVCCCDS Behavior of CCCD Value Store

AT syntax: **AT+LESRVCCCDS=<value>**

*) *Supported since firmware version 5.2*

This command allows to define the behavior of the firmware regarding stored Client Characteristic Configuration Descriptor (CCCD) values.

Value	Properties
0	CCCD values set from bonded peer devices are not stored in non-volatile memory. After reconnect a peer device need to set the CCCD bits again to enable Notifications and Indications.
1	CCCD values set from bonded peer devices are stored in non-volatile memory. After reconnect the GATT Server restores the security level that was required for setting the CCCD before sending Notifications and Indications.

+LESRVBOOTSET Service Set for Boot Activation

AT syntax: **AT+LESRVBOOTSET=<value>**

*) *Supported since firmware version 5.2*

This command allows to define the service set used by **AT+LESRVBOOTMODE** command. At the moment only one service set 1 is supported.

Value	Description
1	Service set is loaded during system startup depending on parameter AT+LESRVBOOTMODE

+LESRVBOOTMODE Boot Behavior of Stored GATT Service Set

AT syntax: **AT+LESRVBOOTMODE=<value>**

*) *Supported since firmware version 5.2*

This command allows to define the behavior of the module during system startup concerning the stored service set.

If you use the **AT+SYSTEMOFF** command it is recommended to use value=2 to avoid unnecessary output during system startup.

Value	Properties
0	Service set is not loaded during system startup. Use AT+LESRVSETACT command to activate service set
1	Service set is loaded during system startup. The channel ID/UUID and the CRC are sent on the serial port
2	Service set is loaded silently during system startup. No output on the serial port

3.4.2. GATT Server Data Handling on AT

+LESRVDATA GATT Server Data Exchange

AT syntax: **AT+LESRVDATA=<channel>,<hexdata>**

*) *Supported since firmware version 5.2*

This command allows the user to set new data on a GATT server characteristic if MUX mode is disabled (AT+BMUX=0).

Channels are created during GATT server definition using the **AT+LEATTRIB** command. After **AT+LEATTRIB=complete**, the GATT server is ready to be used.

All data for defined characteristics is sent over the corresponding channel. If a characteristic has a length of 4, 4 bytes of data has to be sent. The only exception is a characteristic with a variable length.

Parameter	Description
channel	Channel ID from AT+LEATTRIB command
hexdata	ASCII coded byte stream as hexadecimal values e.g. 017aFF for a three byte value. Valid value length: 0 – 20 bytes.

To set new data a larger value than 20 bytes the AT+LESRVDATACONT command can be used.

+LESRVDATACONT GATT Server Data Exchange Continue

AT syntax: **AT+LESRVDATACONT=<channel>,<hexdata>**

*) *Supported since firmware version 5.2*

This command allows the user to set new data on a GATT server characteristic with a data length > 20 byte if MUX mode is disabled (AT+BMUX=0).

Channels are created during GATT server definition using the **AT+LEATTRIB** command. After **AT+LEATTRIB=complete**, the GATT server is ready to be used.

Parameter	Description
channel	Channel ID from AT+LEATTRIB command
hexdata	ASCII coded byte stream as hexadecimal values e.g. 017aFF for a three byte value. Valid value length: 20 bytes.

To set a large characteristic value the data is prepared with one or multiple AT+LESRVDATACONT commands, while the actual write is triggered by a single AT+LESRVDATA command:

```
AT+LESRVDATACONT=<channel>,<hexData>
OK
AT+LESRVDATACONT=<channel>,<hexData>
OK
AT+LESRVDATA=<channel>,<hexData>
OK
```

3.4.3. GATT Server Data Handling on MUX Channel

MUX channels are created during GATT server definition using the **AT+LEATTRIB** command. After “**AT+LEATTRIB=complete**”, the GATT server is ready to be used.

All data for defined characteristics is sent in one MUX frame on the corresponding channel so a MUX frame has to have the defined data length for the characteristic addressed. If a characteristic has a length of 4, 4 bytes of data has to be sent. The only exception is a characteristic with a variable length.

By default the value of a characteristic is empty (length set to zero). It is recommended that the application sets initial data values for all characteristics immediately after registration of all characteristics.

3.4.3.1. Error Handling

All MUX frames with a wrong channel id or a wrong data size are silently discarded.

There is no definition for an error response at the moment in MUX protocol.

Since there is no end-to-end flow control, the GATT server cannot guarantee successful data transfer for notifications and writes without response and will silently discard the data.

3.4.4. Data Handling with or without Connection

All data which is set during a connection is directly sent to the remote side if characteristic definition supports indication or notification. If indications or notifications are not supported only the local server value is updated.

All data which is set without a connection is only updated in the local server.

Only the last value written to a characteristic is stored in the local server.

With every new connection all not signaled data in the server is sent over the link if possible.

4. APPENDIX

4.1. Data Formats

4.1.1. Data Array

Data arrays are encoded as a hexadecimal ASCII based byte stream.

E.g. a Byte array containing the four bytes 0x11, 0x22, 0x33 and 0x44 is encoded as: **11223344**

4.1.2. Bluetooth Address

The BlueMod+S50/Central supports public and random Bluetooth addresses. The differentiation between the address types is done using the parameter “t2” for public addresses and “t3” for random addresses.

A Bluetooth address value itself is a special byte array variant. There are two valid representations.

The Bluetooth addresses “**008025540203**” and **00:80:25:54:02:03** are equivalent.

E.g.	public address:	00:80:25:54:02:03,t2 or 008025540203,t2
	random address:	F1:B9:EB:41:D8:1E,t3 or F1B9EB41D81E,t3

4.1.3. UUID

UUIDs are special byte array variants.

16 Bit UUIDs are encoded with four hexadecimal digits.
E.g. UUID 0xFEFB is encoded as FEFB.

128 Bit UUIDs are encoded with 32 hexadecimal digits.
E.g. 00000002000010008000008025000000.

The format 00000002-0000-1000-8000-008025000000 known from the profile specifications is also supported for 128 bit UUIDs.

4.1.4. Values

A parameter value could be encoded hexadecimal or signed decimal. The value range depends on the command specification.

Hexadecimal values shall be encoded with a leading “0x” e.g. 0x01FF.

Positive decimal values shall be encoded without a leading character e.g. 512.

Negative decimal values shall be encoded with a leading “-“ e.g. -69.

4.1.5. Bit Arrays

Bit arrays shall be coded as hexadecimal values with a leading “0x”.

For an example see parameter PROP in command AT+LESRVD.

4.2. Linktype

Linktype	Meaning
0x02	Bluetooth low energy using public address
0x03	Bluetooth low energy using random address

4.3. AT Result Codes

Result codes (numerical and verbose):

Numeric	Text	Meaning	Extended Result Codes
0	OK	Command completed	No
2	RING	Indicates an incoming call	Yes
3	NO CARRIER	Connection disconnected	Yes
4	ERROR	Illegal command or error	No
9	CONNECT GATT	GATT connection established	Yes
10	CONNECT TIO	TIO connection established	Yes

Extended result codes (numerical and verbose) are available after activation with **ATW1** command.

In multiplexing mode extended result codes are always active.

OK Command Completed

Syntax: OK

Command completed successfully.

CONNECT Connection Established

Syntax: CONNECT connType connHnd [<bdaddr linktype>]

With this result code the user is informed about the establishment of a connection. The connHnd has to be used for characteristic access for this device.

Parameter	Description
connType	Type of connection GATT, TIO
connHnd	Connection handle or TIO MUX channel ID
Bdaddr	Remote Bluetooth address, only as extended result code
Linktype	Remote Bluetooth address type see chapter Linktype, only as extended result code

AT mode: connType represents the connection handle used for **ATH** and GATT client access commands like **AT+LEREAD**, etc..

MUX mode: Additionally, to AT mode description, the value of connHnd represents the channel ID of the automatically established data MUX channel for connections with connType=TIO. Data MUX channels for connections with connType=GATT have to be established using the AT+LEADDCHAN command.

NO CARRIER Connection Disconnected

Syntax: NO CARRIER connHnd [<error code>]

With this result code the user is informed about the disconnection of a connection.

Parameter	Description
connHnd	Connection handle from CONNECT event
error code	Bluetooth release code

RING Link Request Received

Syntax: RING [<bdaddr linktype>]

With this result code the user is informed about an incoming connection request.

Parameter	Description
Bdaddr	Remote Bluetooth address, only as extended result code
Linktype	Remote Bluetooth address type see chapter Linktype, only as extended result code

ERROR Illegal Command or Error

Syntax: ERROR

With this result code the user is informed about an error condition. This could be e.g.:

- an illegal command syntax
- an illegal state for the command
- an error that cannot be indicated otherwise

4.4. Release/Error Codes

The following table shows the release/error codes:

Error code	Meaning
0x0000	Success
0x0001	Accept
0x0002	Reject
0x0003	Resource error
0x0004	Invalid parameter
0x0005	Invalid state
0x0006	Connection disconnect
0x0007	Connection paused
0x0008	Connection lost
0x0009	Authentication failed
0x000A	Flow control violation
0x000B	Init timeout
0x000C	Init out of sync
0x000D	Init hardware failure
0x000E	Lower layer error
0x00FD	Unspecified
0x00FE	Not supported

Examples:

Numerical (ATV0)	3 0x10 <0006>
Verbose (ATV1)	NO CARRIER 0x10 <0006>

4.5. Events

SSPCONF SSP Passkey Confirmation

Syntax: **SSPCONF Bdaddr,tx Passkey ?**

With this event the module requests the confirmation of the passkey displayed on both devices.

Parameter	Description
Bdaddr	Remote Bluetooth address
tx	x is the remote Bluetooth address type (see chapter Bluetooth Address)
Passkey	Passkey to be acknowledged on local side (see **AT+BSSPCONF** command)

SSPPIN SSP Passkey Request

Syntax: **SSPPIN Bdaddr,tx ?**

With this event the module requests the entry of the PIN displayed on the remote device.

Parameter	Description
Bdaddr	Remote Bluetooth address
tx	x is the remote Bluetooth address type (see chapter Bluetooth Address)

SSPPIN SSP Passkey Display

Syntax: **SSPPIN Bdaddr,tx Passkey**

With this event the module shows the PIN to be entered on the remote device.

Parameter	Description
Bdaddr	Remote Bluetooth address
tx	x is the remote Bluetooth address type (see chapter Bluetooth Address)
Passkey	PIN to be entered on remote side

LEATTMTU ATT MTU Updated

Syntax: **LEATTMTU:connHnd,attMtu**

**) Supported since firmware version 5.2*

With this event the user is informed about a ATT MTU update.

Parameter	Description
connHnd	Connection handle from CONNECT event
attMtu	Actual ATT MTU in bytes

LECONPARAM Connection Parameters Updated

Syntax: **LECONPARAM:connHnd,connInt,slaveLat,connTimeout**

With this event the user is informed about a connection parameter update.

Parameter	Description
connHnd	Connection handle from CONNECT event
connInt	Actual connection interval in steps of 1.25 ms
slaveLat	Actual slave latency in connection intervals
connTimeout	Actual connection supervision timeout in steps of 10 ms

LEERROR Error Condition Occurred

Syntax: **LEERROR:type, [parameter]**

With this event the user is informed about error conditions. Depending on error type different parameters are provided.

Type	Parameter	Mode	Description
LEAUTH	connHnd, charHnd, cause	MUX	Error during authentication
LEWRITE	connHnd, charHnd, cause	MUX	Write with response
LEREAD	connHnd, charHnd, cause	MUX	Read
LEMUX	connHnd, charHnd, cause	MUX	Generic error for several situations
LECHANN	channel,cause	MUX	Wrong channel number in command

LEIND Indication Received

Syntax: **LEIND:connHnd,charHnd,<hexData>**

With this event data received over the air with an indication is displayed to the user. To receive this type of event please enable indications with command **AT+LECCCD** if allowed for the characteristic.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values, e.g. 017aFF for a three byte value. Value length: 0 - 22 bytes.

LEINDCONT Indication Received Continue

Syntax: **LEINDCONT:connHnd,charHnd,<hexData>**

*) *Supported since firmware version 5.2*

With this event data received over the air with an indication is displayed to the user. To receive this type of event please enable indications with command **AT+LECCCD** if allowed for the characteristic.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values, Value length: 22 bytes

When the received indication is larger than 22 bytes, the data is split to one or more LEINDCONT and a single LEIND event:

LEINDCONT:connHnd,charHnd,<hexData>

LEINDCONT:connHnd,charHnd,<hexData>

LEIND:connHnd,charHnd,<hexData>

LENOTI Notification Received

Syntax: **LENOTI:connHnd,charHnd,<hexData>**

With this event data received over the air with a notification is displayed to the user. To receive this type of event please enable notifications with command **AT+LECCCD** if allowed for the characteristic.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values, e.g. 017aFF for a three byte value. Value length: 0 - 22 bytes.

LENOTICONT Notification Received Continue

Syntax: **LENOTICONT:connHnd,charHnd,<hexData>**

*) *Supported since firmware version 5.2*

With this event data received over the air with a notification is displayed to the user. To receive this type of event please enable notifications with command **AT+LECCCD** if allowed for the characteristic.

Parameter	Description
connHnd	Connection handle from CONNECT event
charHnd	Characteristic handle from **AT+LESRVD**
hexData	Ascii coded byte stream as hexadecimal values, Value length: 22 bytes

When the received notification is larger than 22 bytes, the data is split to one or more LEINDCONT and a single LEIND event:

LEINDCONT:connHnd,charHnd,<hexData>

LEINDCONT:connHnd,charHnd,<hexData>

LEIND:connHnd,charHnd,<hexData>

LERESOLVED Bluetooth Address Resolved

Syntax: **LERESOLVED:<privacy-bd>,<privacy-bd-type>,<public-bd>, <public-bd-type>**

With this event the user is informed during pairing about a relation between the currently used random resolvable address of a peer device and its public address stored in the bond database.

Parameter	Description
privacy-bd	Current privacy address used by the peer device
privacy-bd-type	Privacy address type (currently only 0x03)
public-bd	Public address of the peer device
public-bd-type	Public address type of the peer device

LESRVDATA GATT Server Data Exchange

Syntax: **LESRVDATA:<channel>,<hexData>**

*) *Supported since firmware version 5.2*

With this event the user is informed about new data on a GATT server characteristic if MUX mode is disabled (AT+BMUX=0).

Channels are created during GATT server definition using the **AT+LEATTRIB** command. After **AT+LEATTRIB=complete**, the GATT server is ready to be used.

Parameter	Description
channel	Channel ID from AT+LEATTRIB command
hexData	Ascii coded byte stream as hexadecimal values, e.g. 017aFF for a three byte value. Value length: 0 - 22 bytes.

LESRVDATACONT GATT Server Data Exchange Continue

Syntax: **LESRVDATACONT:<channel>,<hexData>**

*) *Supported since firmware version 5.2*

With this event the user is informed about new data on a GATT server characteristic if MUX mode is disabled (AT+BMUX=0).

Channels are created during GATT server definition using the **AT+LEATTRIB** command. After **AT+LEATTRIB=complete**, the GATT server is ready to be used.

Parameter	Description
channel	Channel ID from AT+LEATTRIB command
hexData	Ascii coded byte stream as hexadecimal values, Value length: 22 bytes

When the new data on a GATT characteristic is larger than 22 bytes, the data is split to one or more LESRCDATACONT and a single LESRVDATA event:

LESRVDATACONT:channel,<hexData>

LESRVDATACONT:channel,<hexData>

LESRVDATA:channel,<hexData>

4.6. MSCs

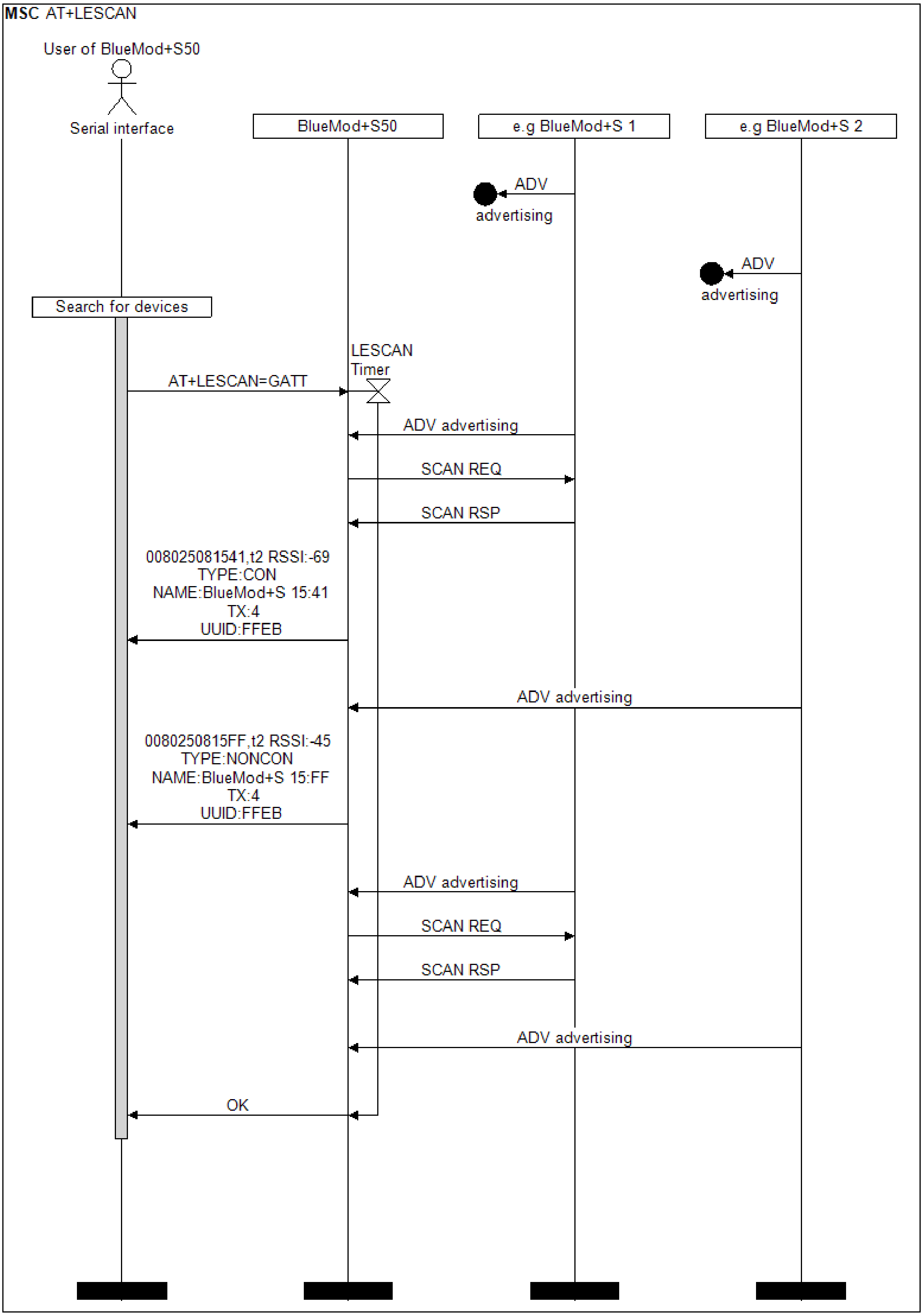

Figure 1: Searching for devices

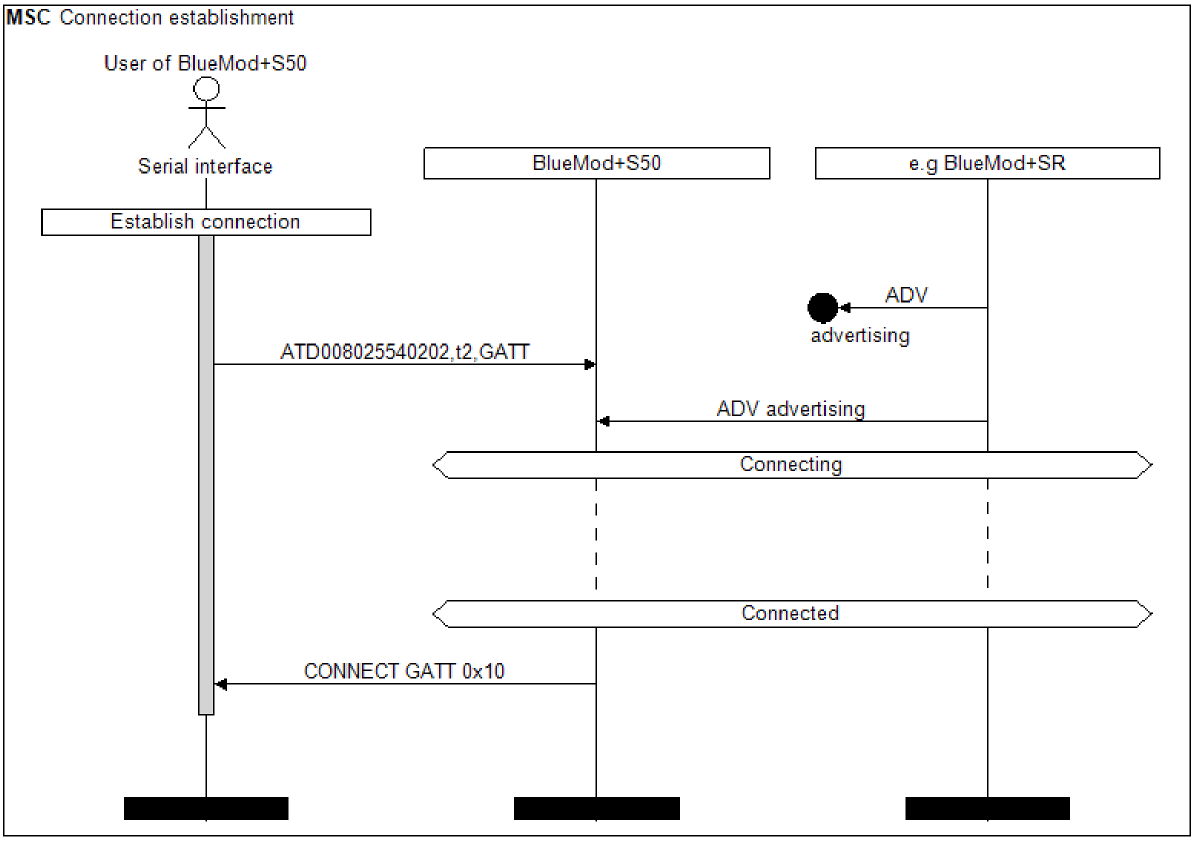

Figure 2: Connection establishment with public type Bluetooth address

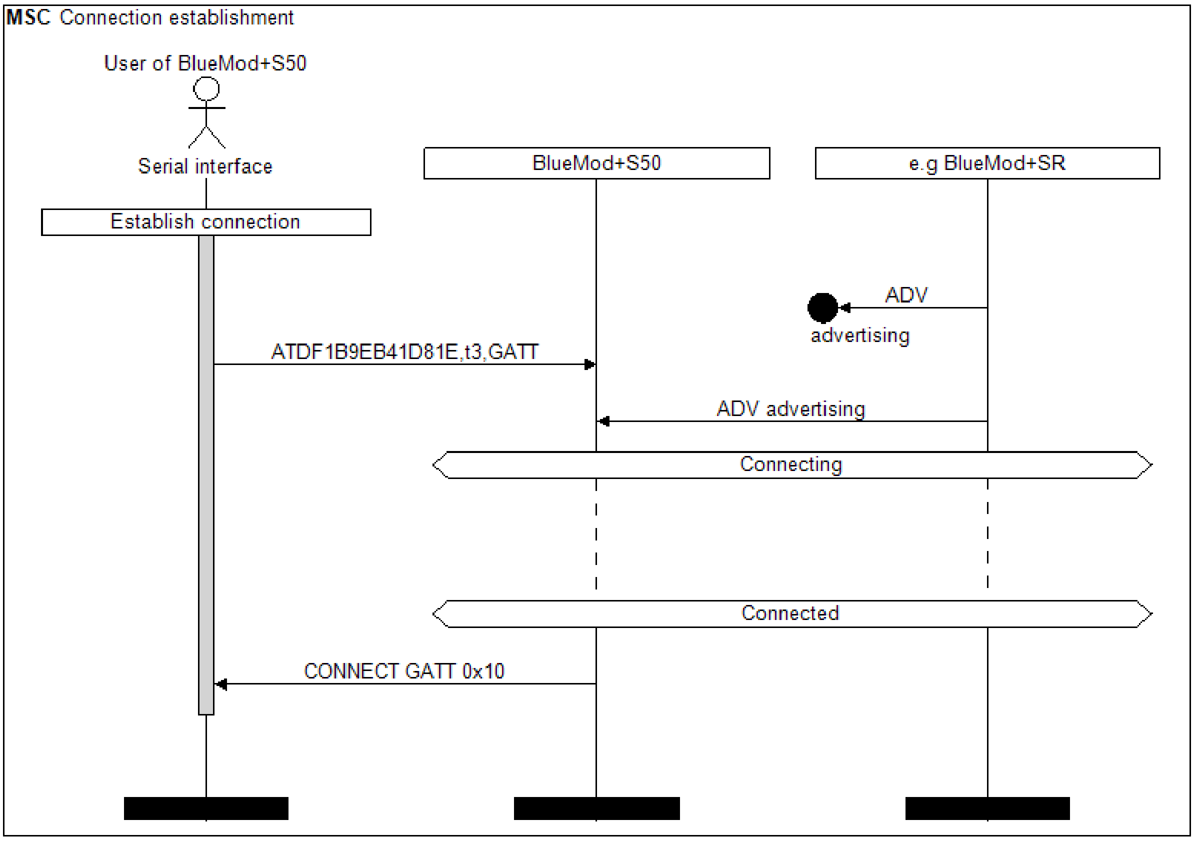

Figure 3: Connection establishment with random type Bluetooth address

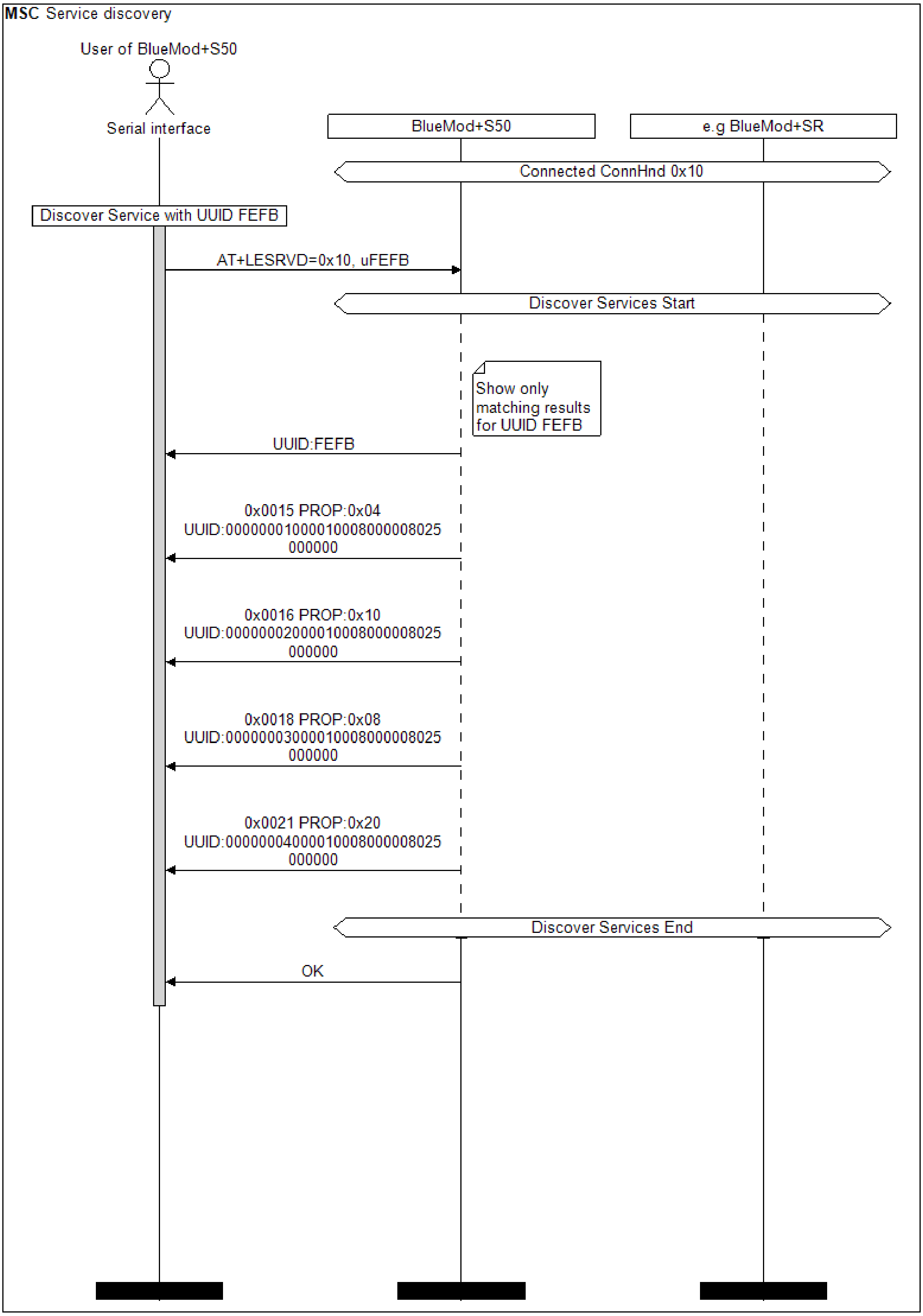
MSC Service discovery
User of BlueMod+S50
Serial interface
BlueMod+S50
e.g BlueMod+SR
Connected ConnHnd 0x10
Discover Service with UUID FEFB
AT+LESRVD=0x10, uFEFB
Discover Services Start
Show only matching results for UUID FEFB
UUID:FEFB
0x0015 PROP:0x04 UUID:00000001000010008000008025 000000
0x0016 PROP:0x10 UUID:00000002000010008000008025 000000
0x0018 PROP:0x08 UUID:00000003000010008000008025 000000
0x0021 PROP:0x20 UUID:00000004000010008000008025 000000
Discover Services End
OK

Figure 4: Service discovery

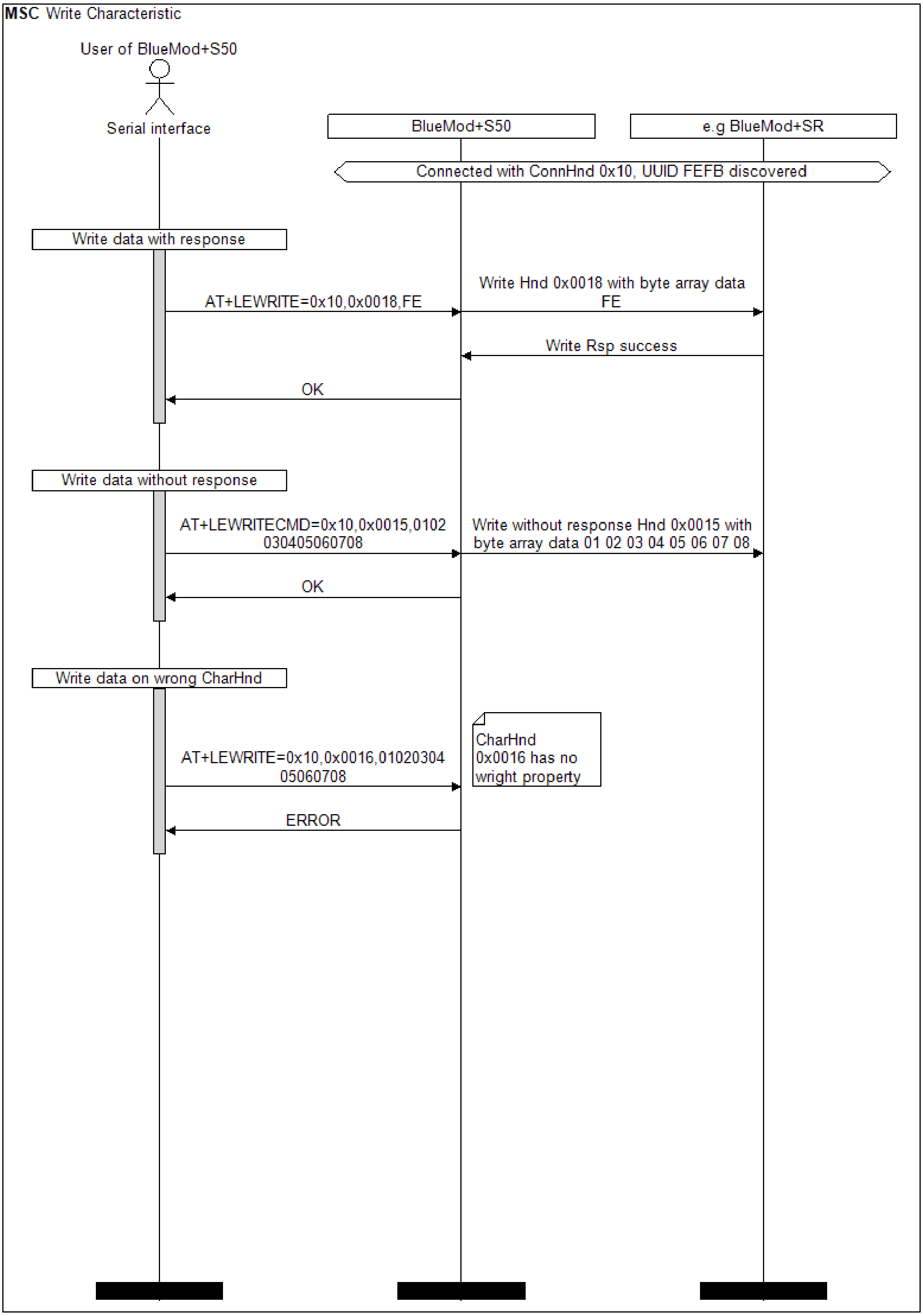
MSC Write Characteristic
User of BlueMod+S50
Serial interface
BlueMod+S50
e.g BlueMod+SR
Connected with ConnHnd 0x10, UUID FEFB discovered
Write data with response
AT+LEWRITE=0x10,0x0018,FE
Write Hnd 0x0018 with byte array data FE
Write Rsp success
OK
Write data without response
AT+LEWRITECMD=0x10,0x0015,0102 030405060708
Write without response Hnd 0x0015 with byte array data 01 02 03 04 05 06 07 08
OK
Write data on wrong CharHnd
AT+LEWRITE=0x10,0x0016,01020304 05060708
CharHnd 0x0016 has no wright property
ERROR

Figure 5: Write characteristic

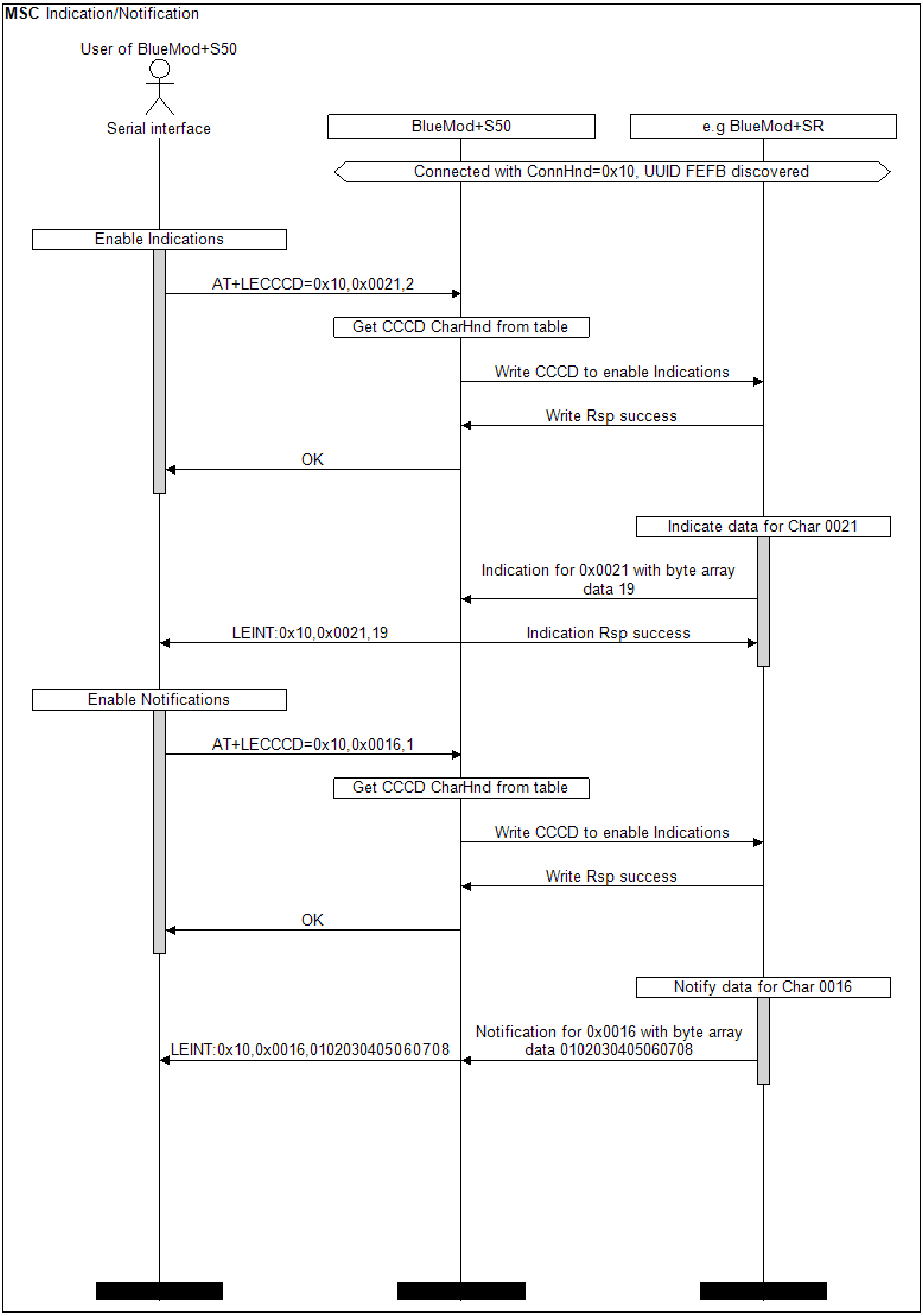
MSC Indication/Notification
User of BlueMod+S50
Serial interface
BlueMod+S50
e.g BlueMod+SR
Connected with ConnHnd=0x10, UUID FEFB discovered
Enable Indications
AT+LECCCD=0x10,0x0021,2
Get CCCD CharHnd from table
Write CCCD to enable Indications
Write Rsp success
OK
Indicate data for Char 0021
Indication for 0x0021 with byte array data 19
LEINT:0x10,0x0021,19
Indication Rsp success
Enable Notifications
AT+LECCCD=0x10,0x0016,1
Get CCCD CharHnd from table
Write CCCD to enable Indications
Write Rsp success
OK
Notify data for Char 0016
Notification for 0x0016 with byte array data 0102030405060708
LEINT:0x10,0x0016,0102030405060708

Figure 6: Indication and notification handling

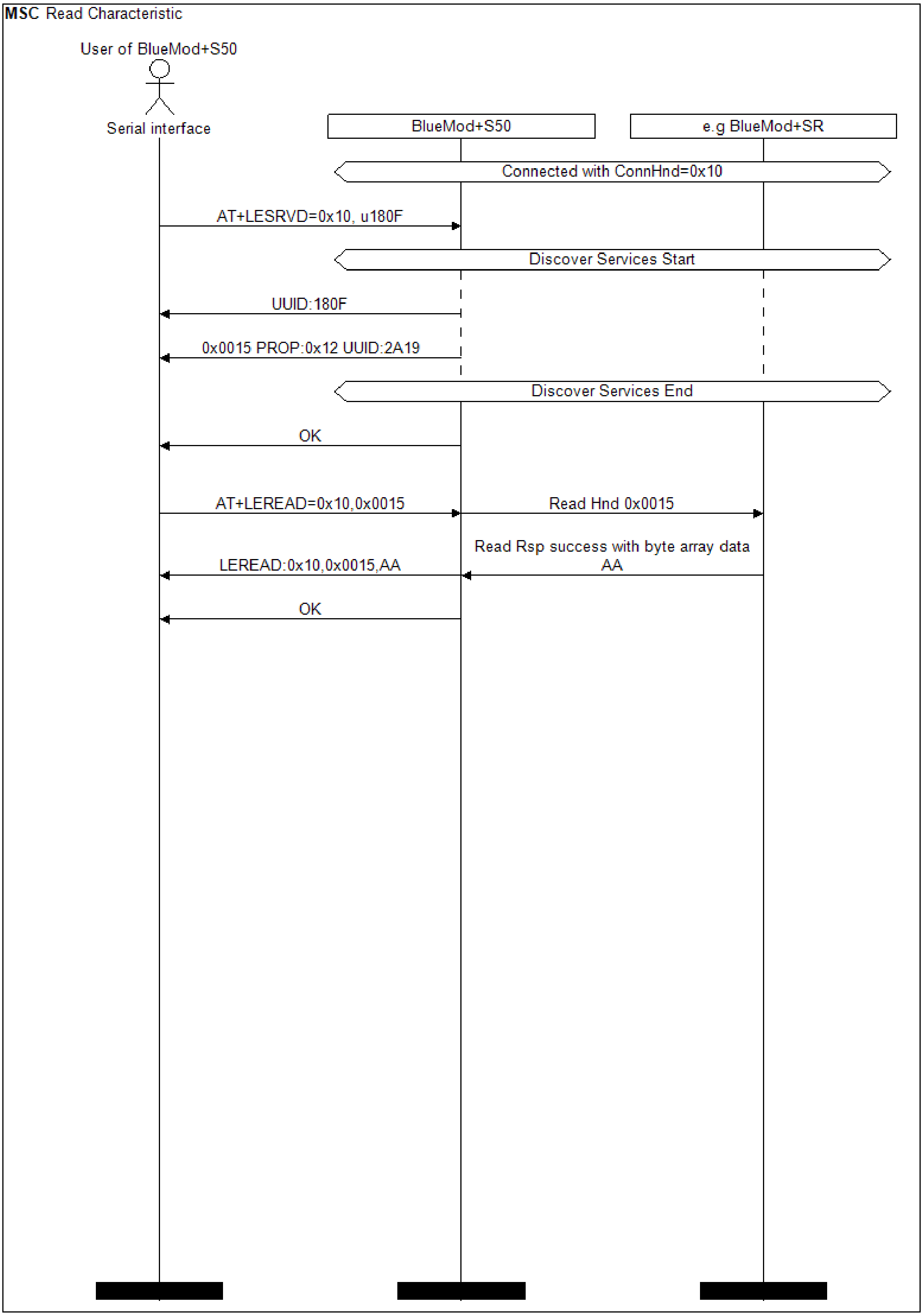

Figure 7: Read characteristic

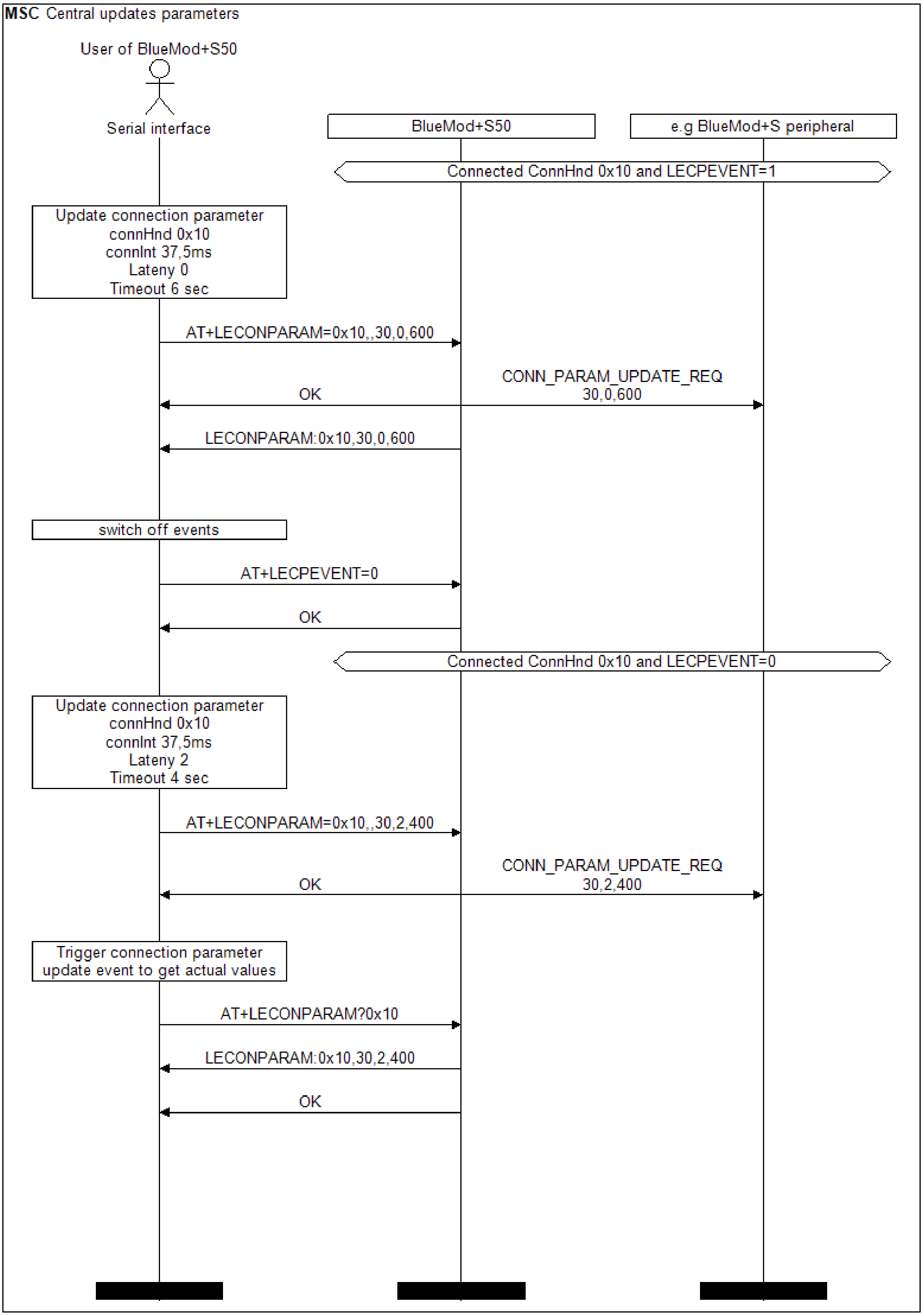

Figure 8: Central connection parameter update

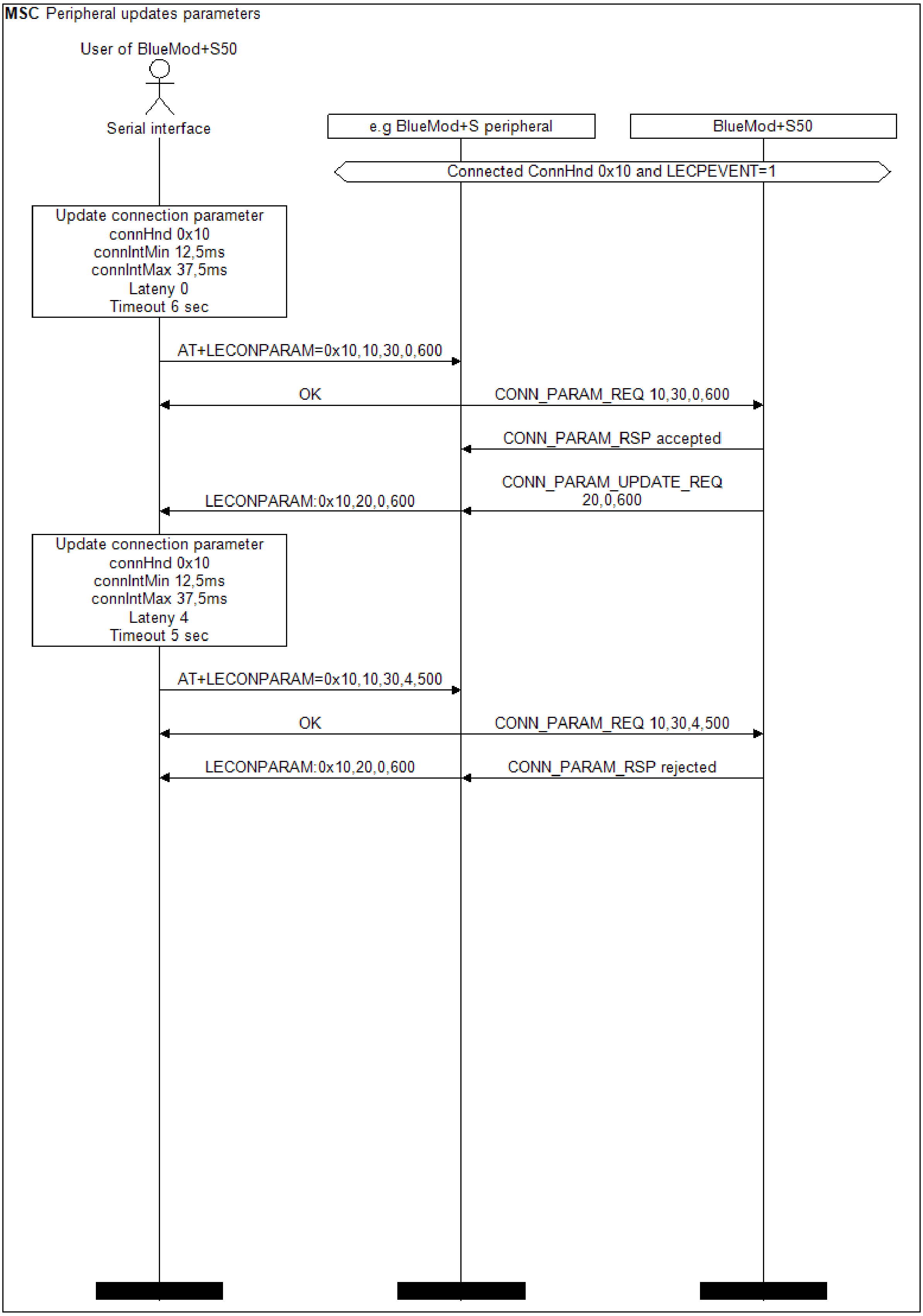
MSC Peripheral updates parameters
User of BlueMod+S50
Serial interface
e.g BlueMod+S peripheral
BlueMod+S50
Connected ConnHnd 0x10 and LECPEVENT=1
Update connection parameter connHnd 0x10 connIntMin 12,5ms connIntMax 37,5ms Lateny 0 Timeout 6 sec
AT+LECONPARAM=0x10,10,30,0,600
OK
CONN_PARAM_REQ 10,30,0,600
CONN_PARAM_RSP accepted
LECONPARAM:0x10,20,0,600
CONN_PARAM_UPDATE_REQ 20,0,600
Update connection parameter connHnd 0x10 connIntMin 12,5ms connIntMax 37,5ms Lateny 4 Timeout 5 sec
AT+LECONPARAM=0x10,10,30,4,500
OK
CONN_PARAM_REQ 10,30,4,500
LECONPARAM:0x10,20,0,600
CONN_PARAM_RSP rejected

Figure 9: Peripheral connection parameter update

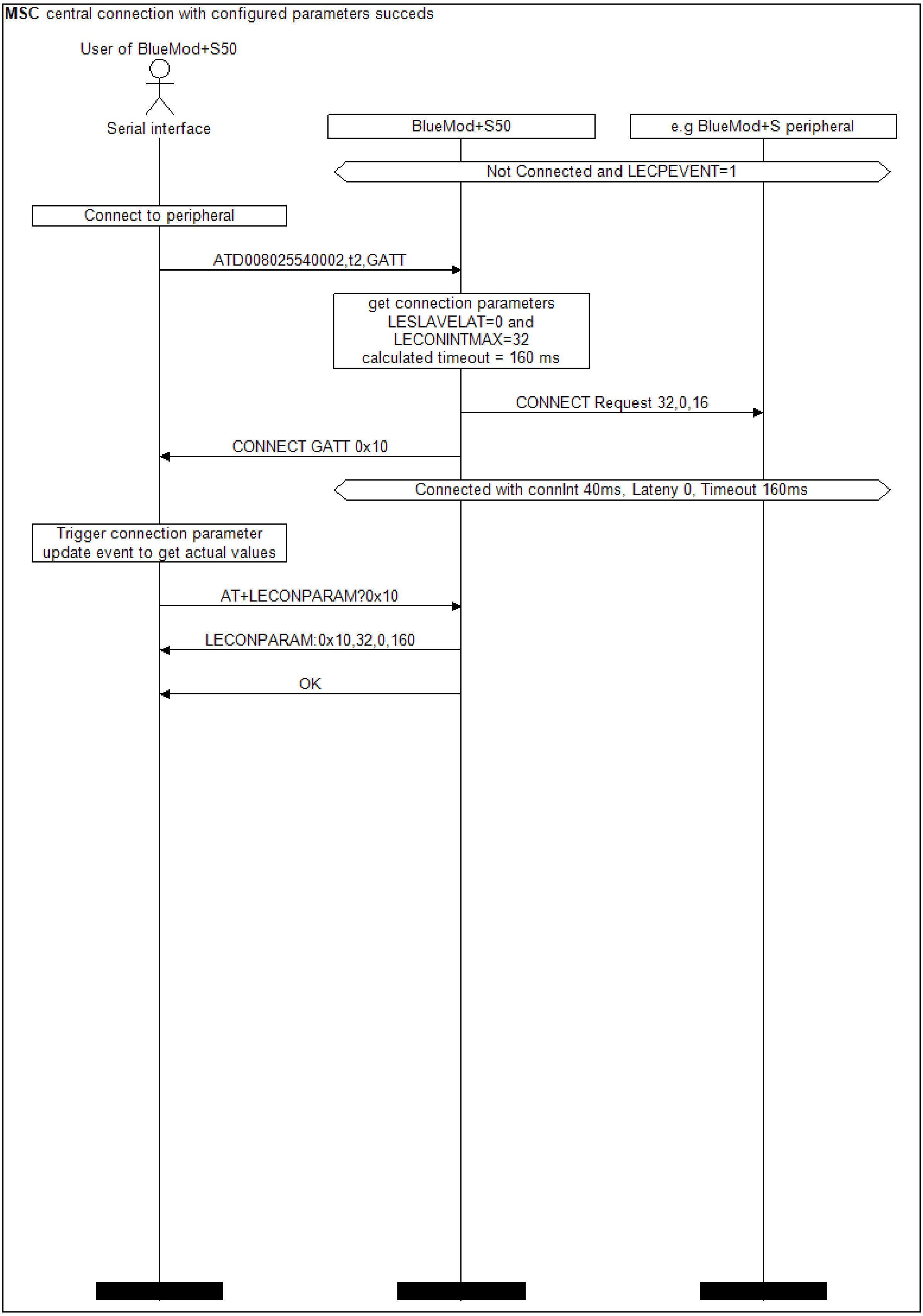
MSC central connection with configured parameters succeds
User of BlueMod+S50
Serial interface
BlueMod+S50
e.g BlueMod+S peripheral
Not Connected and LECPEVENT=1
Connect to peripheral
ATD008025540002,t2,GATT
get connection parameters LESLAVELAT=0 and LECONINTMAX=32 calculated timeout = 160 ms
CONNECT Request 32,0,16
CONNECT GATT 0x10
Connected with connInt 40ms, Lateny 0, Timeout 160ms
Trigger connection parameter update event to get actual values
AT+LECONPARAM?0x10
LECONPARAM:0x10,32,0,160
OK

Figure 10: Central connection parameters during connection establishment

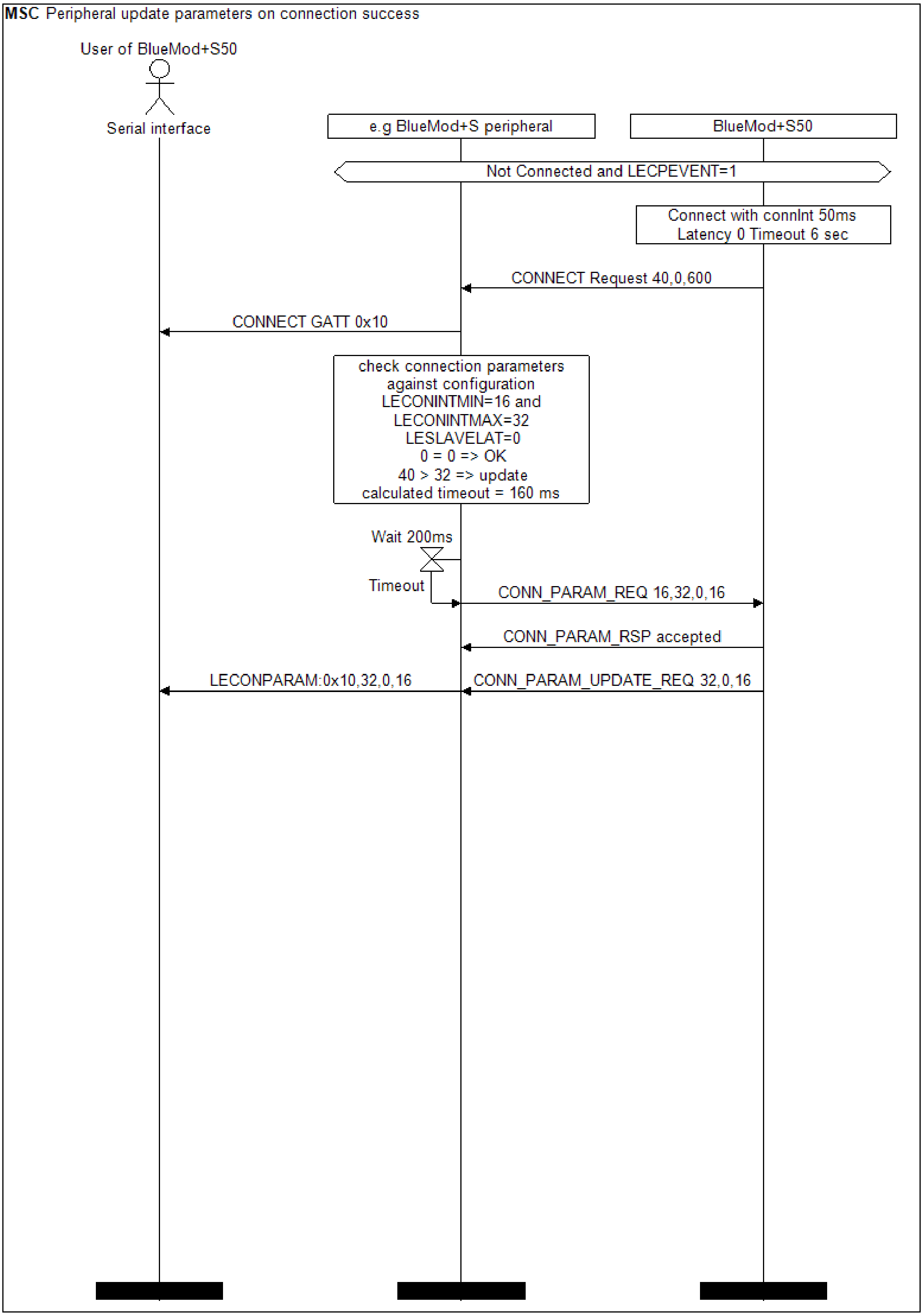

Figure 11: Peripheral connection establishment parameter update success

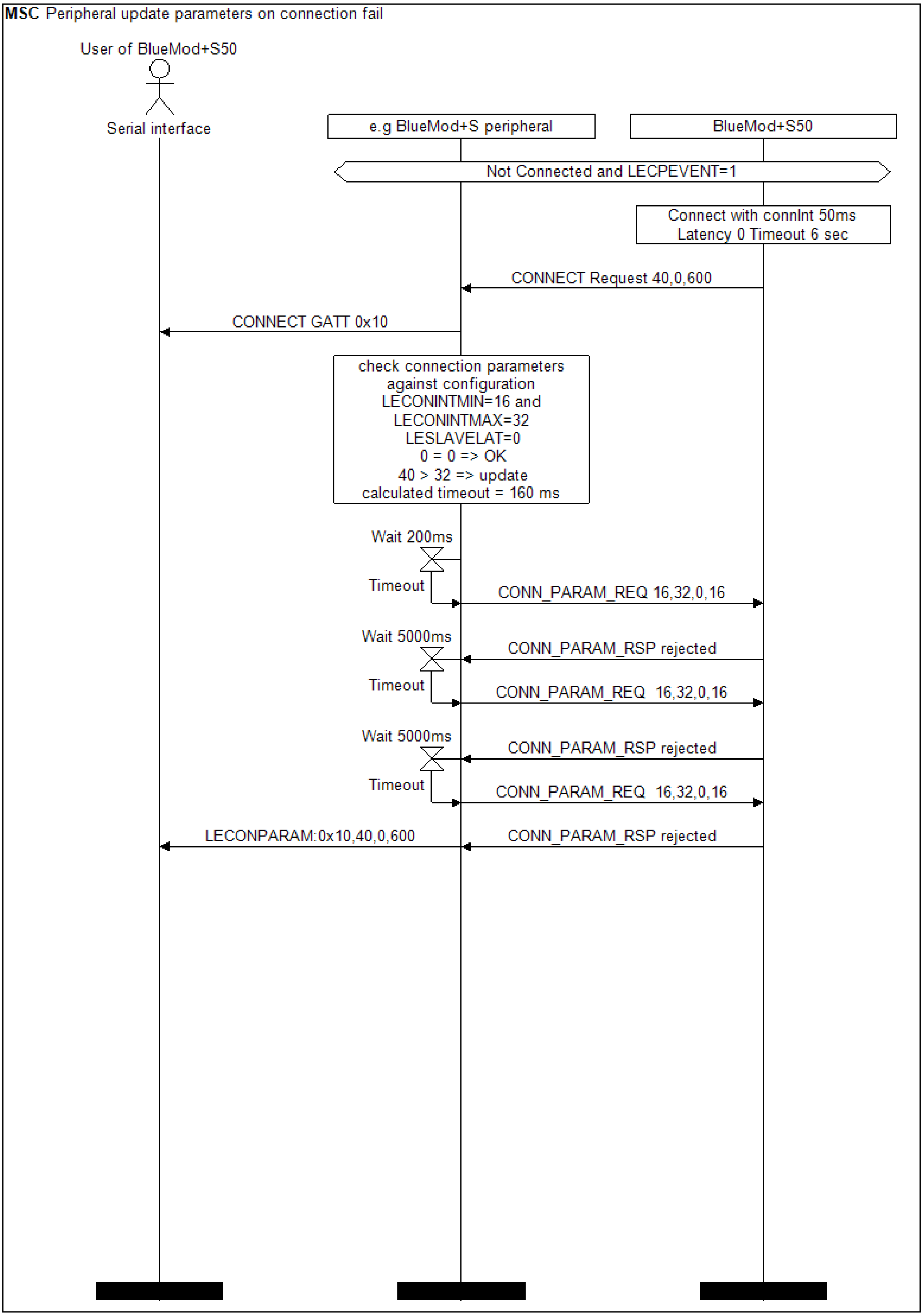

Figure 12: Peripheral connection establishment parameter update fail

5. GLOSSARY AND ACRONYMS

AT	Attention Command
GAP	Generic Access Profile
GATT	Generic Attribute Profile
IRK	Identity Resolving Key
MUX	Multiplexing
SSP	Secure Simple Pairing
UART	Universal Asynchronous Receiver/Transmitter
UICP	UART Interface Control Protocol
UUID	Universal Unique Identifier

6. DOCUMENT HISTORY

Revision	Date	Changes
r0	2018-02-23	First issue
r1	2018-03-26	Added MUX mode Revised description of LEERROR event
r2	2018-09-13	Added generic GATT Client large MTU support (AT+LEATTMTUMAX, AT+LEGATTEVENT, AT+LEWRITECONT, AT+LEWRITECMDCONT, AT+LEATTMTU, LEATTMTU, LEINDCONT, LENOTICONT) Added generic GATT Server support (AT+LEATTRIB, AT+LESRV*, LESRVDATA, LESRVDATACONT) Removed AT+BMITM deprecation note

SUPPORT INQUIRIES

Link to **www.telit.com** and contact our technical support team for any questions related to technical issues.

www.telit.com

Telit Communications S.p.A.
Via Stazione di Prosecco, 5/B
I-34010 Sgonico (Trieste), Italy

Telit Wireless Solutions Inc.
3131 RDU Center Drive, Suite 135
Morrisville, NC 27560, USA

Telit Wireless Solutions Ltd.
10 Habarzel St.
Tel Aviv 69710, Israel

Telit IoT Platforms LLC
5300 Broken Sound Blvd, Suite 150
Boca Raton, FL 33487, USA

Telit Wireless Solutions Co., Ltd.
8th Fl., Shinyoung Securities Bld.
6, Gukjegeumyung-ro8-gil, Yeongdeungpo-gu
Seoul, 150-884, Korea

Telit Wireless Solutions
Technologia e Servicos Ltda
Avenida Paulista, 1776, Room 10.C
01310-921 São Paulo, Brazil